



לוגו בית ספר:

שם בית ספר: איש שלום כפר יונה

שם העבודה: Face Finder

שם התלמיד: אלמוג גל

ת.ז התלמיד: 212645121

שם המנחה: אופיר שביט

שם החלופה: מערכות הפעלה סייבר. הנדסת
תוכנה 883589.

תאריך ההגשה: 3\25\20

עבודת גמר

מסלול 14.50

הגנת סייבר

שנת הלימודים תש"פ

מבנה הספר

1	עמוד שער הפתיחה
2	מבנה הספר
3	מבוא
4	מבנה / ארכיטקטורה של הפרויקט
5	Top-down level design של ארכיטקטורת הפרויקט
6	תפקידי יחידות הפרויקט
8	מבני נתונים שנעשה בהם שימוש
8	זרימת המידע בין היחידות השונות
9	ארכיטקטורת רשת
17	מבנה שרת-לקוח
17	אבטחת מידע של השרת
17	מאגרי מידע
17	אבטחת מידע העובר ברשת
17	תיאור הצפנות
19	תיאור אלגוריתם ראשי
19	הקשרים בין יחידות שונות
19	הצגת מקרים (use cases) עבור פונקציות עיקריות
19	עץ מודולים
21	Use case diagram
21	רשימת Use cases
22	תרשים UML
23	רכיבי ממשק
24	מדריך למשתמש
42	מדריך למפתח
61	בסיס הנתונים
62	ביבליוגרפיה
63	סיכום אישי \ רפלקציה

מבוא

תיק הפרוייקט הזה יעבור על כלל הנושאים המקיפים בהם עוסק הפרוייקט, יעבור על דרך ביצוע אלגוריתמים מסויימים, יעבור על פרוטוקול השרת והלקוח, יעבור על למה בחרתי את הפרוייקט, לאיזה מטרה ובעיה הפרוייקט בא לפתור, סדר התיקיות והצורה שבא הפרוייקט בנוי, קבצים מסויימים בפרוייקט, תהליך מחקר ורקע לפרוייקט, קשיים ופתרונותיהם, רפלקציה אישית שלי, אופן הכנת מסמכים נלווים לפרוייקט, הגבלות ובעיות הפרוייקט וביבליוגרפיה.

כאשר בחרתי לעשות את הפרוייקט הייתי צריך לבחור באיזשהו נושא ורעיון שעליו בעצם יהיה בפרוייקט. בעודי חושב, הסתכלתי על החדשות וראיתי תצלום של מצלמות שמזהות אנשים מסויימים ויודעים מי הם. מאוד התלהבתי מהעניין ולכן בחרתי לקרוא על כך באינטרנט. די הנחתי לא משנה מה הרעיון או הנושא שלי יהיה יש נושאי ליבה שעליהם גם ככה אצטרך ללמוד כגון: בסיס נתונים, שרת לקוח, ממשק משתמש גרפי וכדומה. לכן בנושאים אלו פחות התמקדתי כי בכול מקרה הייתי לומד אותם לא משנה מה היה הנושא שלי. התמקדתי רק על רעיון הזיהוי פנים וזיהוי מי הוא האדם. ראיתי כמה דרגות קושי בין יצירת הכל מהתחלה בעזרת מכונה לומדת, בעצם ללמד מכונה לזהות ולהשוות בין פרצופים, לבין להכין קוד בעזרת מכונה לומדת שכבר אומנה. החלטתי שהחלטת דרגת הקושי בנושא הזה תהיה בסוף הפרוייקט בהתאם לזמן ולכוח שיישאר לי. אבל כן, הנושא סיקרן אותי והתחלתי לקרוא ולראות סרטונים בנושא מכונה לומדת. בסופו של יום, החלטתי שעל זה יהיה הפרוייקט שלי.

תחילה, ביצעתי סקר שוק. כדי לראות האם יש אפליקציות שמבצעות את מה שאני עושה. אם לא, אומנם הפרוייקט יהיה קשה אבל מיוחד וטוב. אם כן, שזה היה המקרה, אז אוכל להיעזר באלגוריתמים של אותה אפליקציה. במהרה מצאתי שישנה אפליקציה די מרכזית בחיינו שמבצעת זיהוי פנים וההשוואה, רק משתמשת בה לדרך קצת שונה וזוהי כמובן גוגל תמונות. כל תמונה שאנו מצלמים בטלפונים החכמים שלנו עולה די אוטומטית לגוגל תמונות, גוגל תמונות מאתר את הפרצופים בתמונה, משווה ביניהם לבין פרצופים שכבר צולמו, ויוצר לך תיקיות של כל הפרצופים שכבר צולמו. לכן הסתכלתי ומצאתי את המכונה הלומדת שבא השתמשה גוגל ותחילה השתמשת בה. האומנם הפרוייקט שלי מסתכל על דברים אחרת, שכן, אני מאתר אנשים מסויימים בתמונות בעוד שגוגל מסדרת תמונות של אנשים בתיקיות. שכן אלגוריתם כמו שלי יש במצלמות אבטחה כדי לאתר פושעים למשל. אזי הפרוייקט שלי די מחדש את השוק כי התקשיתי למצוא איזושהי אפליקציה בשוק שעושה בדיוק מה שהפרוייקט שלי עושה.

התמודדתי עם הרבה בעיות במהלך הפרוייקט שכן עד תחילת עשיית הפרוייקט לא הצלחתי אפילו לכתוב שרת ולקוח. מנגד, החלטתי כן לנסות ולהשקיע המון זמן, בניגוד לקבוצה של תלמידים שפרשו. לאחר למידת הבסיס, טוב ככל שיכולתי, התחלתי לבצע את הפרוייקט. היו לי 2 קשיים מרכזיים בכתובת הפרוייקט: ממשק משתמש גרפי- שימוש בtkinter הקשה עליי מאוד – אבל אם השקעת המון זמן הבנתי את העקרונות והסתדרתי. מכונה לומדת- תחילה הרעיון היה ליצור ולאמן אחת כזאת, אם כי ככל שקראתי על הנושא הבנתי כמה קשה זה יהיה בשבילי והחלטתי להשאיר את זה לסוף.

המוטיבציה שלי להמשיך את הפרוייקט למרות כל הקשיים הייתה מכמה סיבות: אחד נהניתי בהכנת פרויקט באסמבלי בכיתה י' ולכן הנחתי שגם הפרוייקט הזה יהיה מהנה, רצון לעוד 5 יחידות לימוד בתעודת הבגרות מה שיעזור לי בהמשך דרכי, ניסיון וידע שאצבור יעזור לי גם מבחינת פיתוח המוח וגם מבחינת ידע במחשבים, ידע זה יתרום לי לקבלת תפקיד בצבא שקשור למחשבים ובכך יתרום לי לעבוד במקצוע הזה באזרחות דבר שאני מאוד מעוניין בו והסיבה האחרונה שאני מאוד אוהב לאתגר את עצמי גם באתגרים שנראים לי קשים ובלתי אפשריים כדי להראות לעצמי שאני יכול.

הצורך הראשוני שהפרוייקט עונה עליו הוא מציאת איזשהו אדם בין מקבץ תמונות. ז"א יש לפרוייקט המון שימושים למשל צלם אירועים יכול להעלות את כל התמונות שצילם וחפש

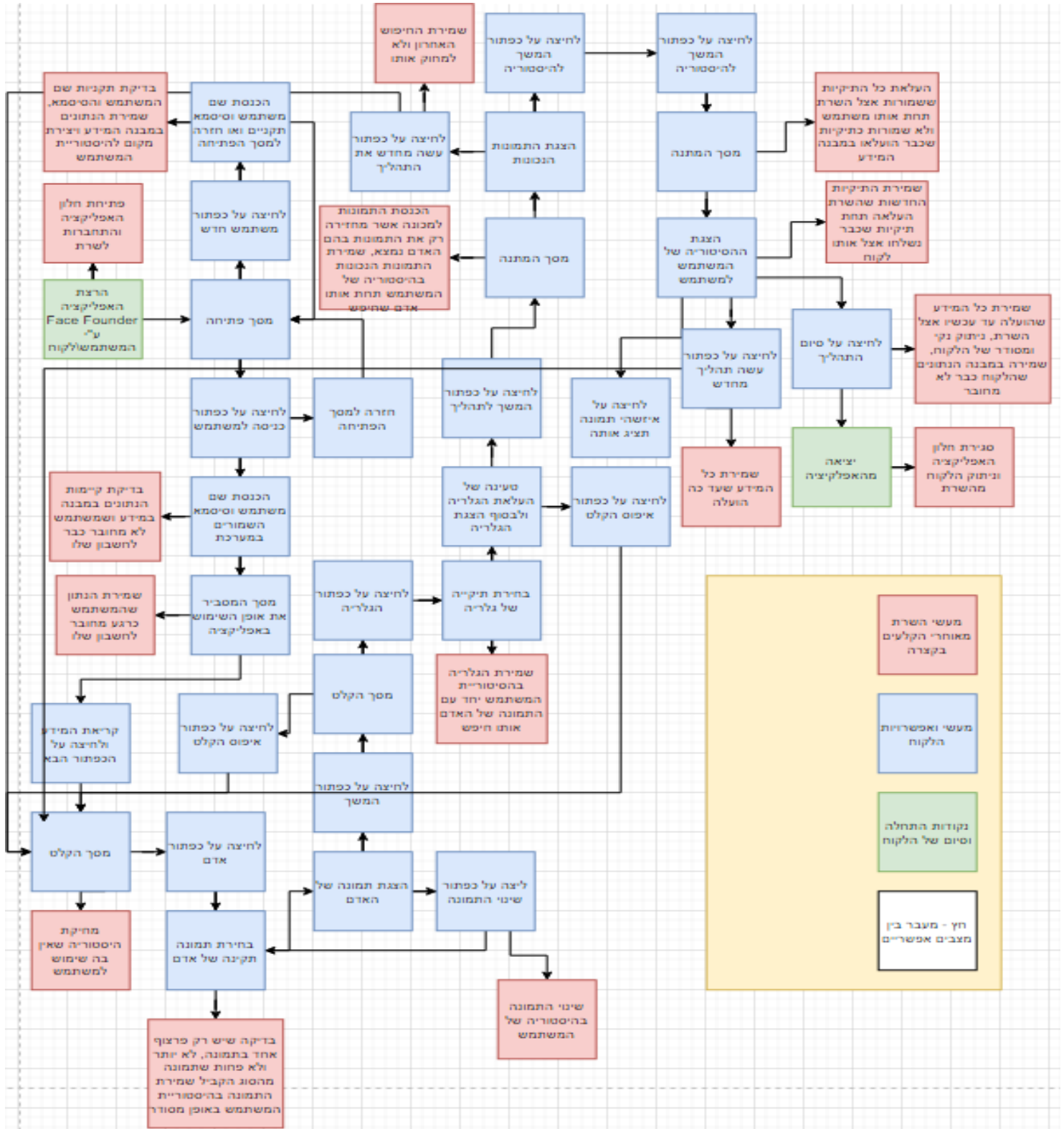
אלמוג גל. Face Founder.

בשביל אנשים תמונות שבהם הם מופיעים (כמובן, שהאפליקציה שומרת את היסטוריית המשתמש) או שימוש אבטחתי כמו למצוא אדם חשוד בין מקבץ תמונות של חשודים. הפתרון הוא מציאת האנשים, תוך כדי שמירת ההיסטוריה של המשתמש באופן מאובטח ומסודר.

מבנה / ארכיטקטורה של הפרויקט

הצגת הפתרון המוצע והסיבות לבחירותיו מוצגות במבוא (עמוד 3, פסקה 2 ו-6).

Top-Down Level Design של ארכיטקטורת הפרויקט:



* כמובן שבכל חלון המשתמש יכול לצאת מן האפליקציה, ולהתנתק מן השרת בצורה לא מסודרת אך נקייה. השרת ישמור במנה הנתונים שהמשתמש כבר לא מחובר לחשבון שלו.

תפקידי יחידות הפרוייקט:

תרחישים עיקריים: בפרויקט ישנם תרחישים עיקריים המחולקים לחלונות של כל חלון יש מטרה מסויימת, ודרכים להגיע לחלונות ספציפיים אחרים. בפרויקט עתיד להיות ממשק משתמש גרפית באמצעות tkinter. התרחישים העיקריים על פי החלונות בפרויקט הם:

הרשמה וכניסה- לאחר הרצת קובץ exe ע"י המשתמש יפתח לו מסך הבית ובו 3 כפתורים: "משתמש חדש", "כניסה".

לחיצה על לחצן "משתמש חדש" תפתח בפני המשתמש חלון ובו 2 תאי מילוי: משתמש וסיסמא. על שם המשתמש וסיסמא להיות באורך לפחות 4 תווים ועל שם המשתמש לא להיות כבר קיים במערכת. לאחר הרשמה תקינה יחזור המשתמש למסך הבית בו יהיה לו רשום "ההרשמה בוצעה בהצלחה. שלום ותודה שהשתמשת ב'זהו פנים'". השרת יבדוק את תקינות הקלט ואחר מכן ישמור את הנתונים התקינים שהמשתמש הכניס בבסיס הנתונים שלו.

לחיצה על כפתור "כניסה" תפתח בפני המשתמש 2 תאי מילוי: משתמש וסיסמא. על שם המשתמש להיות קשור אל הסיסמא ועל שם המשתמש והסיסמא להיות שמורים במערכת. לאחר כניסה תקינה יועבר המשתמש למסך ההוראות. השרת יבדוק שאכן שם המשתמש והסיסמא קשורים זה לזה, שמורים, ושקלט תקין בעזרת בסיס הנתונים. לאחר מכן ישנה בבסיס הנתונים את מצב הסטטוס של הלקוח למחובר.

בכל חלון בסשן זה יהיה כפתור "חזור" שבאמצעותו המשתמש יוכל לחזור למסך הבית המערכת תקפיץ למשתמש הודעה שגיאה בכל פעם שקורה אחת.

הוראות- לאחר ההתחברות של המשתמש אל החשבון שלו יופיע לו מסך הוראות וכפתור "הבא".

במסך ההוראות יהיה רשום למשתמש איך להשתמש באפליקציה:¹העלה תמונה של אדם – תן למערכת Path לתמונה של אדם מסוים. השתדל שהתמונה תהיה ברורה ככל שאפשר. אפשר להעלות רק אדם אחד בכל פעם.²תעלה לאפליקציה תמונות ע"י בחירת תיקיית גלריות מסויימת.³ אם ישנם תמונות אשר הפנים של האנשים בה אינם ברורים סיכויי התמונה למצוא את האדם הרצוי פוחתים.⁴ בסוף התהליך התמונות שימצאו לנכון להיות התמונות בהם האדם נמצא ישמרו באופן אוטומטי במשתמש שלך. בנוסף, תהיינה לך גישה להיסטורי שלך דרך כניסה למשתמש שלך באפליקציה.

לחיצה על כפתור "הבא" תעביר את המשתמש אל מסך קלט.

בנוסף יהיה כפתור חזור שלחיצה עליו תחזיר את המשתמש אל מסך הבית.

קלט- מטרת המסך היא לקבל את המידע הנחוץ לאפליקציה על מנת לפעול- קלט מן המשתמש. בפני המשתמש 3 כפתורים: "אדם", "גלריית תמונות" ו"איפוס הקלט". כמו כן יהיה כפתור "חזור" שיחזיר את המשתמש למסך ה"הוראות". השרת פה "מאחורי הקלעים" מוחק את כל ההיסטוריה הלא נחוצה של המשתמש.

לחיצה על כפתור "אדם" תוביל את המשתמש אל חלון שבו יבחר את האדם ברצונו לחפש. בנוסף על המערכת לבדוק האם הPath קיים, האם כל הקבצים בו מסוג jpg או png, האם המשתמש העלה תמונה בה אין אדם או יותר מאדם אחד. המערכת תודיע למשתמש על כל טעות אפשרית. לאחר הכנסת Path תקין יעבור למסך "אמצע הקלט", בוא תהיה לו האפשרות או לעשות תהליך הקלט שוב ולמחוק את הנוכחי או להמשיך להעלאת הגלריה. השרת יבדוק את תקינות התמונה וישמור את התמונה תקינה תחת ההיסטוריה של אותו משתמש.

לחיצה על כפתור "גלריית תמונות" תוביל את המשתמש לחלון בו המשתמש יבחר איזושהי תיקיית תמונות בה הוא רוצה לחפש את האדם. פה לא תהיינה הגבלה של כמות או של סוג

הקובץ אלא המערכת תודיע מראש שקובץ שאינו מסוג jpg או png, המערכת תבדוק האם Pathn לגלריית התמונות קיים. השרת יבדוק את תקינות הקלט וישמור את הגלריה בהיסטוריה של אותו משתמש תחת אותו אדם.

לאחר העלאת הגלריה המשתמש יעבור למסך "סיום הקלט", בוא תהיה לו האפשרות או לעשות תהליך הקלט שוב ולמחוק את הנוכחי או להמשיך לחלון ההמתנה. "המתנה לקלט" בו יחכה יהיה רשום "חכה בבקשה שהמערכת תסיים לבדוק את תקינות התמונות..."

לחיצה על כפתור "איפוס הקלט" תאפס את כל הקלט ששל המשתמש בתהליך הספציפי הזה ותמחק את התהליך הנ"ל מההיסטוריה של השרת. המשתמש יחזור מחדש למסך ה"קלט".

לאחר הכנסת כל הנתונים יהיה למשתמש כפתור "המשך לתהליך" שלחיצה עליו תוביל למסך "המתנה לסיום תהליך הזיהוי".

המתנה לסיום תהליך הזיהוי- מסך בו יהיה איזשהו עם מדד גרפי בו יהיה רשום כמה תמונות נבדקו עד כה מתוך כמה תמונות, עם הכיתוב "חכה בבקשה שהמערכת תסיים את התהליך..." המשתמש יהיה תקוע במסך זה עד שהמערכת תסיים את התהליך. כאשר התהליך סיים המשתמש יעבור למסך ה"בדיקה וסיום התהליך". השרת ייקח את תמונות הגלריה ואת תמונת האדם ובעזרת מכונה לומדת ימצא באילו תמונות האדם אכן נמצא. לאחר מכן ישמור השרת את התמונות הנכונות ההיסטוריה של אותו משתמש תחת אותו אדם.

בדיקה וסיום התהליך- במסך זה יהיו 2 כפתורים: "עשיית התהליך שוב" ו"לעבור להיסטוריה" ותמונת האדם\אנשים שהמערכת מצאה.

לחיצה על כפתור "עשה את התהליך שוב" תחזיר את המשתמש למסך הקלט הריק ותשמור את החיפוש האחרון שלו בהיסטוריה, אך לא תעלה אותו כי המשתמש לא הגיע למסך ההיסטוריה. לחיצה על כפתור "לעבור להיסטוריה" תעביר את המשתמש ל"מסך ההמתנה להעלאת ההיסטוריה".

מסך ההמתנה להעלאת ההיסטוריה- העלאת ההיסטוריה ללקוח ושמירתה במחשבו תחת pathn של האפליקציות במחשבי windows. השרת יבצע בדיקה בבסיס הנתונים אילו תיקיות היסטוריה השרת כבר העלה, וכך מתוך כל התיקיות שקיימות אצל השרת, הוא ידע אילו תיקיות הוא צריך לעלות. השרת יעלה את התיקיות הרלוונטיות והחדשות ללקוח. לאחר מכן ישמור\יוסיף לבסיס הנתונים תיקיות שכבר הוסיף. כך ידע בפעם הבאה מה לעלות, ובכך השרת מייצל את עבודתו ובאופן כללי את הקוד. לאחר סיום הלקוח יעבור לחלון ה"היסטוריה האישית".

היסטוריה אישית- במסך זה יהיה מוצג במעין תיקיות ששם שם האדם שחיפשת עליו ובכל תיקייה את התמונות שבהם האדם נמצא ותציג תמונת האדם\אנשים שהמערכת חיפשה (בצד בקטן). המשתמש יוכל לעיין בהיסטוריה האישית שלו כרצונו ואף להסיר תמונות. במסך זה יהיה כפתור "קלט נוסף" אשר לחיצה עליו תוביל את המשתמש אל מסך ה"קלט". וכפתור "סיום תהליך ויציאה" אשר יוציא יציאה מסודרת המשתמש מן האפליקציה.

*כל התנתקות תמחק את כל המידע שעוד לא נשמר במערכת ואת כל הקלטים עד כה.

גבולות הפרוייקט- לפרוייקט ישנם את הגבולות הבאים:

הגבלת סוג המידע- הפרוייקט יכול למצוא פנים רק בקבצים מסוגים: jpg, png. ועל כן ישנה הגבלה מבחינת המשתמש שצריך לדאוג שהמידע שהוא מעביר לאפליקציה תואם בסוגו.

הגבלות כוח זיכרון- מכיוון שתיכנון data basen שלי צורך הרבה זיכרון, יכול להיווצר מצב של עודף מידע בdata base, כמובן מבחינה תאורטית אם לאפליקציה תהיה רב משתמשים.

אלמוג גל. Face Founder.

הגבלות כוח מעבד - האפליקציה תפעל עם threads כך שתוכל לבצע כמה פעולות במקביל, ובכל לטפל בכמה לקוחות במקביל ולא תיתקע בכל בקשה שדורשת כוח מעבד ובכך זמן ריצה רב. ולכן, שוב מבחינה תאורטית, האפליקציה תפעל לאט יחסית משימוש רב משתמשי כי היא מוגבל בכוח מעבד.

הגבלות מהירות - יתכן שבגלל הסיבות שציינת לגבי כוח הזיכרון והמעבד יחכו המשתמשים זמן מה בין חלונות, ולכן יש לי גיבוי בחלון ה"המתנה".

הגבלת תוכנה – האפליקציה תעבוד רק למשתמשי windows.

מבני נתונים שנעשה בהם שימוש:

בפרויקט, בעיקר מענייני נוחות השתמשתי ברשימות, גם כי אני כותב בפייתון, שבו רשימות הם הדבר היותר מקובל (בניגוד לשפות C למשל). בנוסף גם במבנה -חיה סדורה (tuple). אם כי, ישנם פעמים שהשתמשתי ברשימה שבעצם דימתה מבנים אחרים כמו למשל רשימה שדימתה מחסנית. דוגמא לשימוש רגיל של רשימה הוא לשמור ברשימה את כתובת הלקוח יחד עם מפתח ההצפנה שלו, בחיה סדורה, כל חיבור של לקוח חדש. כך השרת ידע באיזה אופן להצפין לכל לקוח. דוגמא לשימוש של רשימה שדימתה מחסנית היא כאשר לקחתי כל התיקיות של אותו לקוח תחת אדם מסוים, מכיוון שהפקודה של הוצאת תיקיות מpath מסוים מסדרת את התיקיות לפי זמנים, בעצם כמו במחסנית, האחרון שהוכנס הוא במעלה המחסנית (LIFO), יכולתי לדעת שהתיקיה הראשונה ברשימה תהיה התיקיה עם התמונות הנכונות של אותו אדם, כאמור תמיד תיקיה זו תיווצר אחרונה.

זרימת המידע בין היחידות השונות:

בעצם, נקטתי בשיטה של סדר שנשמרת לאורך כל ה"מסע" של הלקוח. כל נתון רלוונטי של הלקוח ישר נשמר אצל השרת. כך לא יאבד מידע. ולאחר מכן, כל מה שנשאר זה תיאום, בעצם, שהשרת תמיד "ידע" איפה ב"מסע" כל לקוח נמצא. הסדר די כרונולוגי בפרויקט, חוץ מכמה לולאות דבר שמאפשר לשרת לדעת שהמידע תמיד באיזה סדר מסוים אותו הוא יודע, ומפשט דברים ללקוח.

ארכיטקטורת הרשת:

פרוטוקולי הרשת: ההודעות ברשת מועברות באמצעות מערכים, למען נוחות. לכן כל הודעה עוברת pickle ומתורגמת בעצם מרשימה להודעת string וכמובן בצד השני גם להפך (loads\dumps).

הודעות הלקוח לשרת:

הסבר	שם משתמש ממי נשלחה ההודעה	מידע נחוץ נוסף	מידע נחוץ	הבקשה/הפעולה	סוג ההודעה
בקשה ליצור משתמש חדש עם הסיסמא 1234 מאת המשתמש Almog	Almog		1234	CNU (create new user)	re (request)

re (request)	LTA (login to account)	1234		Yossi	בקשה להיכנס אם הסיסמא 1234 אם המשתמש של Yossi
ok (waiting)		NP (no password)		Almog	הודעת שנועדה שהלקוח Almog ישלח הודעה כדי שיהיה מוכן לקבל הודעה מהשרת, ולהודיע לשרת על תפקוד הלקוח Almog
re (request)	exit	NP (no password)		Almog	הודעה של הלקוח על יציאת המשתמש Almog מהאפליקציה
re (request)	IPI (insert person image)	מידע של התמונה מקריאה בינארית שלה	Trump.png	Almog	בקשת הכנסת תמונה למשתמש של Almog אדם- Trump אותו יחפשו, מידע של התמונה, באמצעותו השרת ישמור אצלו את התמונה, ושם התמונה- Trump.png
re (request)	IGF (insert gallery folder)	My Gallery	Almog		בקשת יצירת גלריה בשם My Gallery אצל הלקוח Almog

re (request)	IPTG (insert person to gallery)	מידע של התמונה מקריאה בינארית שלה	Moshe.jpg	Almog	בקשת הכנסת תמונה של Moshe תחת השם Moshe.jpg לגלריה האחרונה העכשווית אצל הלקוח Almog השרת בודק גם את תקינות התמונה, סוג ושיש בה פרצוף אחד.
re (request)	FTP (find the person)	NP (no password)		Almog	בקשת מציאת האדם העכשווי בגלריית התמונות של האדם הנ"ל של הלקוח Almog
re (request)	SNHF (send new history folders)	NP (no password)		Almog	בקשה לשלוח תיקיות ההיסטוריה שעוד לא שלחת ללקוח Almog
re (request)	SITH (send images to history)	NP (no password)		Almog	בקשה לשלוח תמונות לתיקיות ההיסטוריה החדשות תוך כדי שתשלח לי איזה תמונה לאיזה תיקייה, ללקוח Almog

re (request)	GPK (get public key)	NP (no password)		NU (no username)	בקשה לשליחת מפתח ההצפנה הגלוי
re (request)	CASK (create a symmetrical key)	NP (no password)	14641	NU (no username)	בקשה שהשרת יצור מפתח סודי ואישי ללקוח שממנו נשלחה ההודעה בעזרת המספר 14641 שנוצר. בעזרת RSA

* כמובן כאשר אין מידע נחוץ נוסף הפאקטה בגודל 4 ואין בה תא ריק.

* שליחת ראשי התיבות של ההודעות כמובן, מוסכמת על השרת והלקוח, נועדה לחסוך גודל של מידע ועומס מיותר על השרת וגם נועדה ליצור עוד איזושהי רשת ביטחון כאשר ידוע שההודעה בין 3 ל 4 תווים.

* ברוב המקרים השרת מחזיר תשובה בהתאם להודעה של הלקוח.

הודעות השרת ללקוח:

הסבר	מי נשלחה ההודעה	מידע נחוץ נוסף	מידע נחוץ	הפעולה שבוצעה	סוג ההודעה
השרת אישר את יציאת הלקוח – יציאה נקייה ומסודרת	Server			exit	CO (confirmed)
שליחת המפתח הגלוי	Server	145678	5	PKS (public key sent)	CO (confirmed)

CO (confirmed)	SKS (symmetrical key saved)	5	145678	Server	הודעה שהמפתח הסימטרי סודי אישי ללקוח נשמר בהצלחה. בנוסף מצורף המפתח הציבורי להודעה.
ER (error)	NU (no username)			Server	השרת מודיע על שגיאה בשל אי הכנסת שם משתמש בעת יצירת משתמש חדש
ER (error)	NP (no password)			Server	השרת מודיע על שגיאה בשל אי הכנסת סיסמא בעת יצירת משתמש חדש
ER (error)	UTS (username too short)			Server	השרת מודיע על שגיאה בשל הכנסת שם משתמש קצר מדי בעת יצירת משתמש חדש
ER (error)	PTS (password too short)			Server	השרת מודיע על שגיאה בשל הכנסת סיסמא קצרה מדי בעת יצירת משתמש חדש

ER (error)	UAE (username already exists)			Server	השרת מודיע על שגיאה בשל הכנסת שם משתמש שכבר קיים בעת יצירת משתמש חדש
CO (confirmed)	NUC (new user created)			Server	השרת מודיע ללקוח שהחשבון נוצר בהצלחה
ER (error)	UOPW (user or password are wrong)			Server	השרת מודיע ללקוח על שגיאה בשל הכנסת שם משתמש אולגם סיסמא שגויה
ER (error)	AAIU (account already in use)			Server	השרת מודיע ללקוח על שגיאה בשל ניסיון להיכנס לאותו חשבון במקביל
CO (confirmed)	PGI (person got inserted)			Server	השרת מודיע ללקוח שהתמונה של האדם שהעלה נשמרה ושנפתחה לו תיקיית היסטוריה בהצלחה

CO (confirmed)	GGC (gallery got created)			Server	השרת מודיע ללקוח שיצירת תיקיית הגלריה לאדם אותו הוא מחפש נוצרה בהצלחה
CO (confirmed)	PITG (people inserted to gallery)			Server	השרת מודיע ללקוח שהעלאת הגלריה, של האדם אותו הוא מחפש, בוצעה בהצלחה
CO (confirmed)	PITI (person in this image)	Trump.jpg	מידע של התמונה מקריאה בינארית שלה	Server	השרת מודיע ללקוח שהאדם שחיפש נמצא בתמונה Trump.jpg מתוך הגלריה שהלקוח העלה לאותו אדם. האדם הוא האדם העכשווי, האחרון שהועלה.
CO (confirmed)	NMIF (no more imaged found)			Server	השרת מודיע ללקוח שלא נמצאו יותר תמונות בהם מופיע האדם אותו הלקוח חיפש בגלריה שאותו צירף לאותו אדם

CO (confirmed)	HFS (history folders send)			Server	השרת מודיע ללקוח ששלח את תיקיות ההיסטוריה החדשות, שהשרת לא העלה לפני.
CO (confirmed)	IITH (insert image to history)	Trump	Person two more places in array: Trump.jpg מידע של התמונה מקריאה בינארית שלה	Server	השרת מודיע ללקוח שישנה עוד תמונה שיש להעלות להיסטוריה של אותו לקוח. עליו להעלות בתיקייה של Trump בתת התיקייה Person את התמונה Trump.jpg וזאת יעשה באמצעות המידע של התמונה
CO (confirmed)	NMHF (no more history files)			Server	השרת מודיע ללקוח שסוימה ההעלאה של ההיסטוריה, אין יותר

					קבצי היסטוריה חדשים
--	--	--	--	--	------------------------------------

* כמובן כאשר אין מידע נחוץ נוסף הפאקטה בגודל 4 ואין בה תא ריק.

* שליחת ראשי התיבות של ההודעות כמובן, מוסכמת על השרת והלקוח, נועדה לחסוך גודל של מידע ועומס מיותר על השרת וגם נועדה ליצור עוד איזושהי רשת ביטחון כאשר ידוע שההודעה בין 3 ל4 תווים.

* ברוב המקרים השרת מחזיר תשובה בהתאם להודעה של הלקוח.

* השרת שולח את ההודעות שלו באופן מאובטח, באמצעות הצפנה כך שרק השרת והלקוח אליו הייתה אמורה להישלח ההודעה ידעו מה תוכן ההודעה ולא גורם שלישי עוין.

מבנה שרת-לקוח:

שרת ראשי, שאליו ניתן להתחבר כמה לקוחות במקביל. השרת יודע לטפל בכל לקוח בנפרד, ולדעת לאיזה לקוח שייך איזה מידע. שימוש במודול select.select לבד מבחין בין לקוחות שהתנתקו, ללקוחות שהתחברו, לקוחות מחכים להודעה ובעזרתו אפשר לשלוח כל הודעה ללקוח הנכון.

אבטחת מידע של השרת:

השרת שומר מידע פנימי של הלקוח, כגון הסיסמא שלו, ההיסטוריה האישית שלו וכדומה אצלו במחשב בתיקייה אישית שאין אליה גישה מבחוץ, אלא רק השרת\המחשב ממנו מורץ השרת יכול לגשת לתיקייה. בתיקייה זו נשמרים שני דברים: הראשון, מידע רב שלא ניתן לקשה לשמור אותו בבסיס נתונים מסוג sqlight3. זה כולל תיקייה לכל לקוח שם כל הקבצים שהעלה אותו לקוח נשמרים, מסודרים, נמחקים אם אין להם שימוש, ונעשה בהם שימוש של השרת. הדבר השני הוא database מסוג sqlight3 בו נשמרים דברים שלא נכון לשמור אותם בתיקיות ועליהם להיות שמורים גם במקרה שהשרת נופל לדוגמא. כגון שם משתמש וסיסמתו. כך בעצם כל המידע הנ"ל שמור ונגיש רק לשרת. לא ניתן באף דרך לפרוץ למחשב השרת וגנוב מידע.

מאגרי מידע:

כאמור, כל מאגרי המידע נמצאים על מחשב השרת. בתיקייה על מחשב השרת. תיקייה לכל לקוח בה יש תיקייה לכל אדם שאותו לקוח חיפש. בה יש 3 תיקיות: האדם המבוקש, גלריית תמונות, ותמונות בהם האדם נמצא. ובסיס נתונים בו יש את כל לקוח את שם המשתמש שלו, הסיסמא שלו, הpath של השרת בו נמצא המידע של הלקוח, האם הוא מחובר ותיקיות היסטוריה שהשרת כבר העלה ללקוח.

אבטחת מידע העובר ברשת:

קיים מידע רגיש שאסור שגורם עוין שלישי יקבל העובר ברשת. לשם כך קיים צורך שרק השרת והלקוח המסוים אליו מתכוון השרת לשלוח הודעה ידעו את המידע הנשלח. לשם כך היה צורך חיוני להצפין את המידע ברשת, בדרך שרק 2 הגורמים שצוינו קודם יוכלו לפענח אותו. החלטתי להשתמש בהצפנת RSA, אותה מימשת בעצמי. בנוסף עוד הצפנות משנה. כל ההצפנות מפורטות בהמשך.

תיאור הצפנות:

RSA: כאשר השרת נהיה אקטיבי, הוא קורא לפונקציה שיוצרת את המפתח הציבורי. המפתח הציבורי נשמר כמשתנה גלובלי לאורך כל זמן הרצת השרת.

המפתח הציבורי בנוי משני מספרים: מספר ראשון - **n** - מכפלת שני מספרים ראשוניים רנדומליים. **m** - ביצוע פונקציית אולר (Euler) על המספר הראשון. מספר שני - לאחר יצירת **m** התחלה של חיפוש מספר החל מ-5 (כל מספר ראשוני קטן עובד, בחרתי ב-5) של מספר שגם לא זוגי וגם זר לחלוטין (אין להם מחלק משותף) מציאת המספר הנ"ל הינו המספר השני - **e**. הפתח הציבורי, כשמו הוא, נשלח לכל לקוח שמעוניין בו. לאחר מכן השרת יוצר מפתח פרטי שרק הוא יודע, זאת באמצעות אלגוריתם פשוט לשרת, אך לניחוש פורץ מאוד מסובך. אני בחרתי ליצור מספר **k** שיהיה שווה לאחד ויגדיל את עצמו באחד בלולאה. בכל פעם בלולאה אני קורא לפונקציה **be** אני מכפיל את **k** ב**m** מוסיף להם אחד ומחלק ב**e**. אם התוצאה שלמה היא המספר הפרטי של השרת - **d**.

כאשר לקוח מעוניין להתחבר הוא מקבל את המפתח הציבורי. לאחר מכן רק לקוח שיוזע את הפרוטוקול של השרת (אם כי עדיין בעל סיכוי לניחוש\פריצה) ידע שהוא צריך ליצור בעצמו עוד מספר. יצירת המספר של הלקוח היא יצירת מספר רנדומלי (באמצעות Random) בין המספר הציבורי השני שקיבל לבין מספר היכולת הרצה של המחשב שלך (ככל שהכוח המעבד שלך קטן יותר כך המספר יקטן). מכיוון שאני עובד במחשב ביתי, ואין ברשותי כוח מעבד כמו חברות אבטחה גדולות שמגיעות גם למספרים של 50 ו-60 ספרות, מספר ההגבלה שלי הוא 150000. הגעתי למספר זה באמצעות לולאה ששיניתי כל פעם את המספר עד שהגעתי למספר ההגבלה הגדול ביותר וזמן ההרצה הקטן ביותר (לא יותר מכמה שניות במקרה הגרוע ביותר). לאחר יצירת המספר הנ"ל הלקוח מעלה את המספר שיצר בחזקת המספר הראשון הציבורי והשאירית של התוצאה מהמספר השני הציבורי תיתן את המפתח הסימטרי הפרטי. הלקוח שולח את המספר הרנדומלי (לא הפרטי) שמצא לשרת.

השרת מקבל את המספר מעלה אותו בחזקת המספר הפרטי של השרת. לתוצאה הוא מחפש שארית מהמספר השני הציבורי. והתוצאה היא המפתח הסימטרי הפרטי. השרת שומר ברשימה את הלקוח ואת המפתח הסימטרי פרטי שלו, כמובן לכל לקוח מפתח סימטרי פרטי אחר. כך יודע השרת כשאר הוא מדבר עם לקוח איך להצפין ולפענח הודעות שנשלחות ביניהן כך שרק הם ידעו.

כך, בלי לשלוח ברשת את המפתח שבו יצפינו הלקוח והשרת באותה שיטת הצפנה (שאפרט בהמשך) ללקוח ולשרת יש "ביד" את אותו מספר. כך מובטח שגם אם גורם שלישי יאזין ברשת, בלי המספר הפרטי של השרת **d** הוא לא יוכל לדעת מהו המפתח הציבורי הפרטי של הלקוח והשרת.

הצפנת משנה: לאחר שלשרת יש מפתח סימטרי פרטי אם כל לקוח יש להסכים שצד הלקוח והשרת לגבי אלגוריתם הצפנה ופיענוח שרק מחזיק מפתח מסוים ידע מה המידע בהודעה. אלגוריתם ההצפנה בנוי כך: פונקציה המקבלת מפתח מסוים, ופאקטה (הודעה) מסוג רשימה. הפונקציה עוברת על כל הודעה ברשימה בנפרד ומצפינה אותו בנפרד.

הודעה מוצפנת כך: עוברים על כל האותיות של ההודעה. לכל אות מעבירים למספר Unicode שלו (באמצעות פונקציית ord). את המספר הנ"ל הופכים (נגיד 1546 הופך ל6451). מחסירים ממנו 100 עד שנמצא בין 0 ל255, בינתיים ישנו מונה (ערכו הראשוני 40) שעולה, בכל החסרה של המספר ההפוך, במפתח ההצפנה. מכניסים לתוך רשימה את המספר ההפוך ואת המונה. לאחר מכן עוברים לאות הבאה. בסופו של דבר ישנה רשימה המייצגת מילה עם מספר הפוך ומונה וכך הלאה. לאחר מכן, כל איבר ברשימה משונה למספר ascii שלו. כך בעצם לא ניתן לדעת ברוב מוחלט של המקרים איזה מספר ייצג התוצאה כי המספר המקורי היה גדול שלא ניתן כמעט לשחזר אותו אחורנית להיות מספר. לאחר מכן, מחברים לstring אחד את כל הרשימה וכל הרשימה הנ"ל מייצגת מילה אחת בפאקטה. כך עוברים על כל המילים בפאקטה ומצפינים את כולם. רק למי שיש את המפתח

יהיה מסוגל להעביר את האותיות ascii למקור, שכן הוא יודע כמה (בעזרת צירוף המונה ליד כל אות) פעמים להכפיל את המפתח ואיתו להחזיר לאחורנית לאות. פונקציית הפיענוח פשוט הפוכה מפונקציית ההצפנה וכך בעצם מפענחים ומצפינים חזרה למקור את כל ההודעות בלי למחוק מידע כלל. בעצם גורם המסתכל בפאקטות העוברות ברשת יראה רק מידע מוצפן. גם בעזרת brute force לא ימצא מהי הפאקטה המקורית כי אין ברשותו את המפתח. כאמור הוא יכול לנחש פעם את המפתח ועליו לנסות brute force, אבל כמובן דבר זה ייקח לו שנים. חיסרון יחיד קטן, שבו נתקלתי והצלחתי לעקוף זה שאתה חייב לשלוח תווים שיש להם Unicode. זה הפריע לי כאשר ניסיתי לשלוח מידע של תמונה שבנוי מאלפי תווים אבל זוהי בעיה ידועה באינטרנט אליה מצאתי פתרון. בנוסף לכל הודעה, בגלל שהיא רשימה, עוברת pickle, שהופך אותה לstring ארוך. דבר שמקשה עוד יותר על גורם שלישי עוין.

וכך, מועברות באופן מאובטח לחלוטין תחילה באמצעות RSA מפתחות סימטריים פרטיים ואז באמצעות אלגוריתם ההצפנה הנ"ל מידע ופאקטות רגישים וסודיים. בעצם רק השרת והלקוח יודעים את ההודעות שהם שולחים זה לזה.

תיאור אלגוריתם ראשי:

שרת: מורץ בלולאה תמידית ומחכה לחיבור של לקוחות, אותם הוא שומר. השרת רק מקבל הודעות ופועל בהתאם. השרת מנווט את המידע שלו בין הלקוחות ושולח להם אותו, או מקבל אותו בהתאם ועושה פעולות הקשורות ללקוח בהתאם. בנוסף השרת תומך בלקוח הכי "טיפש" שיכולתי לבנות, ז"א שרוב ענייני האלגוריתם בקשר לתמונות, מכונות וכו' נמצא אצל השרת.

לקוח – תפקיד האלגוריתם אצל הלקוח הוא לדאוג לזרימה מסוימת של הלקוח באפליקציה על פי סדר כרונולוגי. למשל, לנווט את השרת בין חלון לחלון הבא המתאים לו, לדעת שבתחילת הקוד יש לדאוג שהלקוח יבקש מפתח ציבורי וכדומה. השרת מנווט כך את הלקוח שרואה זאת ויזואלית באמצעות GUI שגם האלגוריתם שלו שייך לשרת. בעזרת ה-GUI הלקוח גורם למשתמש לנוע ברחבי באפליקציה, באופן מסודר ושניתן לפקח עליו ולהראות לו מידע וחלונות בהתאם. השתמשתי ב-GUI שלי tkinter. בנוסף, באמצעות ה-GUI הלקוח מקבל קלט מהמשתמש, שומר ומעבד אותם אצלו ושולח בהתאם הודעה לשרת, ואז מציג את תשובת השרת. הלקוח דואג גם להציג תמונות ללקוח, וכך בעצם גם לשמירת מידע, תמונות, על מחשב הלקוח באופן הכי יעיל ושלא ישימו אליו לב.

הלקוח הוא בעצם זה שדואג לעניינים החזותיים והנוחות בעוד שהשרת על האלגוריתמיקה הטכנית.

הקשרים בין יחידות שונות: ניתן לראות תחת Top-Down-Level-Design של ארכיטקטורת הפרויקט.

הצגת מקרים (use cases) עבור פונקציות עיקריות:

ניקח שתי פונקציות עיקריות בקוד שלי:

1. הכנסת האנשים המתאימים המקבלת את שם המשתמש, שם התמונה, ומידע בינארי של התמונה.

```
def insert_correct_images(username, name, data):
```

קיים משתנה גלובלי בתוכנית שלי אותו אני מאפס אחרי כל תהליך והוא מערך של התמונות הנכונות (correct_images). הפונקציה הנ"ל יוצרת path במקום שנועד לאפליקציות windows בשם של השעה והתאריך המדויק. לאחר מכן באותו path יוצאת קובץ התואם לשם הקובץ ושומרת אותו. הפונקציה כותבת בינארית בו את כל המידע שקיבלה. הפונקציה סוגרת את הקובץ ושומרת את התמונה במערך התמונות הנכונות. לסיום שולחת לשרת

שקיבלה את התמונה שבה האדם נמצא, ושהיא מחכה לעוד תמונות. ראה דוגמא תחת use case diagram, עמוד 20.

2. פונקציית "חלון הקלט". פונקציה שמקבלת את שם המשתמש, חלון הרקע - שערכו הדיפולטיבי הוא תמונה של חלון הקלט, משתנה בוליאני אדם - שערכו הדיפולטיבי שקר ומשתנה בוליאני גלריה - שערכו הדיפולטיבי שקר.

```
def input_window(username, background=Inputs_B,
person=False, gallery=False):
```

הפונקציה מוחקת את כל הכפתורים והרקעים שהיה בהם שימוש קודם לכן, כדי לא לבזבז מקום מיותר ועבודת מעבד מיותרת. הפונקציה מציגה את הרקע של התמונה שקיבלה, הרקע הראשוני של חלון הקלט או רקע במקרה שהוא כבר בחר אדם או כבר בחר אדם וגלריה. הכותרת של החלון תהיה Input Window.

כאשר משתנה האדם הוא שקר סימן שהמשתמש לא הכניס אדם. לכן יופיע לו כפתור של "בחר אדם". אליו יכנס ויבחר אדם. בחלון נפרד. או שילחץ על כפתור אתחול הקלט ויחזור לתחילת תהליך חלון הקלט.

כאשר המשתנה אדם נכון, סימן שהמשתמש כבר בחר אדם ולכן הרקע שיוצג לו הוא שכבר בחר אדם. בנוסף אם אדם סימן שלא נבחרה עוד גלריה ולכן יופיע לו כפתור "בחר גלריה". אליו יכנס ויבחר גלריה. בחלון נפרד. או שילחץ על כפתור אתחול הקלט ויחזור לתחילת תהליך חלון הקלט, שגם אדם הוא שקר.

לאחר מכן אם גלריה נכון, וגם אדם נכון יוצג רקע שהמשתמש סיים את תהליך הקלט. יופיע לו כפתור "המשך לתהליך" שלחיצה עליו תמשיך בתהליך מציאת הפנים. בחלון נפרד. או שילחץ על כפתור אתחול הקלט ויחזור לתחילת תהליך חלון הקלט שגם אדם שקר וגם גלריה שקר.

בנוסף לאורך כל המצבים ישנה דאגה לevent בו המשתמש סוגר את החלון, מצב יציאה, שיקרה לפונקציית יציאה שתדאג ליציאה מסודרת.

עץ מודולים:

Server_Functions.py:

```
import select
import pickle
from Server.Preparation_For_Encryption import *
from Server.Encryption_And_Decryption import *
import base64
import os.path
import os
import matplotlib.pyplot as plt
import datetime
import ntpath
# connection setting:
from Server.Communication_Settings import *
import socket
import glob
from Server.Detect_faces import *
import shutil
```

Preparation_For_Encryption.py:

אלמוג גל. Face Founder.

```
from random import randrange, getrandbits
import time
```

Detect_Faces.py:

```
from __future__ import print_function
import click
import os
import re
import face_recognition.api as face_recognition
import multiprocessing
import itertools
import PIL.Image
import numpy as np
import ntpath
```

Database.py:

```
import sqlite3
import os
import psutil
```

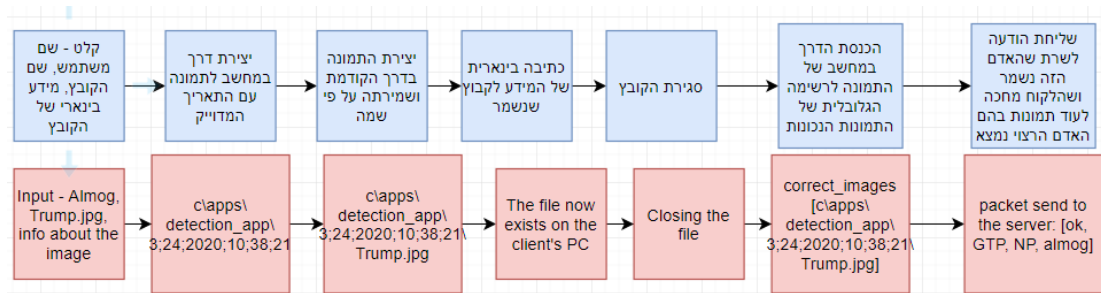
Handle_Images.py:

```
import sqlite3
import os
import psutil
```

Client_Functions.py:

```
from tkinter import DoubleVar, filedialog, ttk
#StringVar
import socket
import pickle
from Client.Encryption_And_Decryption import *
import random
import base64
import ntpath
import matplotlib.pyplot as plt
import datetime
from PIL import Image, ImageTk
import time
# connection setting:
from Client.Communication_Settings import *
from Client.Face_Detection_And_Comparison import *
from Client.Handle_Images import *
import shutil
```

:Use case diagram



רשימת use cases

ישנם שלושה חלקים עיקריים לפרויקט: פתיחה- הרשמה וכניסה למערכת, הכנסת קלט של המשתמש, הצגת פלט למשתמש. ניתן use case לכל אחד מהחלקים האלו:

1. משתמש לוחץ על כניסה למערכת ומופיע לפניו שתי שורות לכתיבה. אחת שם המשתמש ואחת הסיסמא. נניח שהמשתמש טועה בכתיבת הסיסמא שלו ולוחץ על כפתור "הבא" לוחץ על כפתור המקלדת enter. המידע יועבר מן הGUI אל הלקוח. הלקוח ישלח את שם המשתמש והסיסמא בפאקט לשרת. ההודעה תוצפן והשרת יפענח אותה. השרת יבדוק ברשימת הsql שלו האם ישנו אותו שם משתמש שהוכנס תחת העמודה "שם המשתמש". כשאר ימצא, דבר שיקרה כי המשתמש טעה בסיסמתו, השרת יעבור לבדוק מהי הסיסמא המתאימה לשם המשתמש שמצא, ואותה ישווה לסיסמא שקיבל מהלקוח. כאשר ישווה ויראה שהסיסמא שנשלחה ע"י הלקוח לא תואמת לסיסמא בטבלת הsql תחת אותו משתמש יוציא השרת הודעת error. בהודעה ירשום השרת ללקוח ששם המשתמש או הסיסמא שגויים. ההודעה תוצפן ותפוענח ע"י הלקוח. הלקוח יראה שההודעה מוסרת שישנה טעות אחד מהנתונים שהכניס הלקוח שגויים. לכן הלקוח יחליף את הרקע של מסך הכניסה למשתמש, במסך (שעוצב מראש על ידי) שבו רשום בהודעה אדומה ששם המשתמש או הסיסמא שגויים. השרת ישמור את שם המשתמש וסיסמא השגויים ויכניס אותם לשורות בהתאם, למקרה למשל כמו זה, בו ישנה טעות קטנה בהקלדת הסיסמא. כך הלקוח ידע על הטעות וגם לא יצטרך לרשום את פרטיו מחדש אלא פשוט לתקן את הטעות בסיסמתו, להזין אותה שנית, וכל התהליך חוזר, עד אישור השרת לכניסת המשתמש למערכת.

2. משתמש מגיע למסך הקלט. תחילה עליו להכניס תמונה של אדם אחד (לא יותר ולא פחות) כקלט שאותו ירצה למצוא לאחר מכן בתוך גלריית תמונות. המשתמש בטעות לחץ על התמונה הלא נכונה, ולחץ על התמונה שבה הרבה אנשים. התמונה תחילה, אצל הלקוח, ממורת למידע בינארי. המידע הנ"ל נשלח לשרת תחת בקשה של יצירת תיקייה חדשה לאותו אדם, אליה יועלה מידע בהמשך. המידע מוצפן ומפוענח ע"י השרת. השרת ממיר את המידע הבינארי ושומר את התמונה שנשלחה אצלו בשרת. השרת מעביר את הpath של התמונה למכונה הלומדת שבדקת האם יש פרצוף אחד בתמונה. המכונה תחזיר שבתמונה יש כמה פרצופים. השרת ישלח הודעת error ללקוח שבתמונה יש יותר מפרצוף אחד. ההודעה תוצפן ותפוענח ע"י הלקוח. הלקוח יקבל את ההודעה וישנה את מסך הרקע בחלון הקלט של האדם אותו תרצה לחפש, לרקע שבו צצה הודעת שגיאה לפיה בתמונה יש יותר מפרצוף אחד, ועל המשתמש לחפש אדם אחד. המשתמש יראה את ההודעה (כמו כן, תנאי זה הופיע לו בעמוד ההוראות שעבר בו קודם לכן), ויבין שלחץ על התמונה הלא נכונה. תהייה לו האפשרות לבחור תמונה שוב, והפעם יבחר המשתמש בזירות את התמונה אליה התכוון מלכתחילה. התהליך חוזר, והפעם השרת יאשר את התמונה שהועלתה והלקוח יוכל להמשיך במסך הקלט שלו (יצטרך להכניס גלריה).

3. המשתמש סיים את התהליך. כרגע מוצגות בפניו התמונות בהם האדם שחיפש נמצא ושתי אפשרויות, בנוסף המידע של התהליך הזה נשמר כבר אצל השרת. כפתור "לעשות את התהליך מחדש" אשר לחיצה עליו תגיע ללקוח והוא יחזיר את הלקוח למסך הקלט. וכפתור "עבור להיסטוריה". לחיצה על כפתור זה תגרום ללקוח לשלוח בקשת העלאת קבצי היסטוריה לשרת. ההודעה תוצפן ותפוענח ע"י השרת. השרת יקבל את הפאקטה, בה רשומים פרטי המשתמש. באמצעותם ילך לשרת המשתמש בטבלת sql ושם לעמודת תיקיות היסטוריה. השרת יכניס לרשימה את כל התיקיות הנ"ל. השרת יקבל את path של הלקוח גם בעזרת פרטיו וטבלת sql שברשותו. כך ידע לאן לגשת ולהוציא את כל תיקיות ההיסטוריה שקיימות לאותו משתמש במחשב לו. תיקיות אלו מעולות ישר לשרת, אך לא ישר ללקוח. בעזרת חיסור הרשימות ידע השרת אילו תיקיות עוד לא העלה ללקוח ואת שמם ישלח ללקוח עם הוראה ליצור תיקיות אלו בpath ההיסטוריה של הלקוח. ההודעה תוצפן ותפוענח ע"י הלקוח. הלקוח באמצעות משתנה גלובלי יודע איפה ההיסטוריה של המשתמש שמורה אצלו במחשב. שם, לפי השמות שקיבל, ייצור תיקיות היסטוריה בהתאם. לאחר סיום ייצור ההיסטוריה ישלח הלקוח שסיים לעלות את תיקיות ההיסטוריה. ההודעה תוצפן ותפוענח ע"י השרת. השרת יקבל את ההודעה ויתחיל לשלוח קבצי היסטוריה רלוונטיים. השרת דאג לזכור את התיקיות החדשות, וקודם שומר שהעלה אותם בטבלת sql תחת אותו שם משתמש תחת עמודת תיקיות היסטוריה. לאחר מכן עבור על קובץ בתיקיות הנ"ל שולח השרת את המידע של הקובץ, ואיפה לשמור אותו ללקוח. לאחר מכן מחכה שנייה אחת כדי שהלקוח יעמוד בקצב. הלקוח כל הודעה מקבל ומפענח וכך הוא יודע איזה תמונה ואיפה לשמור בהיסטוריה. לאחר העלאת כל התיקיות מוצגת ספריית ההיסטוריה למשתמש, שם הוא יוכל לעיין, למחוק, להעתיק וכו' קבצים.

תרשים UML:

Class: DATABASE

Operations	Values	מה הפעולה עושה
Database	db_location	פותחת קובץ מסוג db. הקובץ הינו מבנה נתונים, טבלה מסוג sql, שניתן לשמור בה מידע
fetch		מחזירה את כל הנתונים בטבלה.
insert	username, password	מחזירה אמת אם לא קיים כזה שם משתמש ושם המשתמש והסיסמא הוכנסו בהצלחה, אחרת שקר.
exists	username, password	מחזירה אמת אם קיים שם משתמש שתואם לסיסמא, אחרת שקר
generate_path	username	יוצרת path להיסטוריה של המשתמש ושומרת אותו, ומחזירה אותו
active_user	username	אם משתמש לא אקטיבי, הופכת אותו לאקטיבי ומחזירה אמת, אחרת שקר

unactive_user	username	הופכת משתמש ללא אקטיבי, אם הפעולה בוצעה בהצלחה מחזירה אמת
remove_user	username	מוחקת מהטבלה שם שורה שבה נמצא שם המשתמש
unactive_all		הופכת את כל המשתמשים בטבלה ללא אקטיביים
update_user	username, password	מעדכנת את הסיסמא של שם המשתמש, במידה וקיים
running		מחזירה true אם הטבלה מורצת מ cmd או מ python
get_user_path	username	מחזירה את path ההיסטוריה של שם המשתמש
insert_user_history_folder	username, folder_name	מוסיפה ושומרת שם נוסף של תיקיית ההיסטוריה בעמודת "תיקיות ההיסטוריה"
get_history_folders	username	מחזיר את כל תיקיות ההיסטוריה ששמורות תחת שם המשתמש
del		מוחקת את כל הטבלה

רכיבי ממשק:

ממשק המשתמש הגרפי, GUI, נכתב כולו ב Tkinter .tkinter הינו ממשק משתמש גרפי בשפת python באמצעותו ניתן לעצב חלונות, כפתורים, רקעים, תאי מילוי, פס גלילה, מד טעינה, הצגת תמונות ועוד. מצאתי זאת לנוח לכתוב את כל הפרוייקט בשפה אחת – python ולכן נבחר. רוב הלקוח שלי הוא קוד tkinter. בעיצוב האפליקציה הושקע הרבה שורות קוד, חשיבה וזמן.

מדריך למשתמש:

על המשתמש להוריד ספריית קבצים בשם "Client" לאותו path במחשב שלו. שם יהיו לו 6 קבצים/תיקיות:

תיקייה ראשונה- Design – שם יהיו כל עיצובי הרקע והGUI של האפליקציה.

קובץ Client_Functions.py – שם יהיו כל פונקציות הפייתון הקשורות לתפעול הלקוח, קובץ הלקוח המרכזי שבסופו של דבר מורץ.

קובץ Communication_Setting.py – קובץ שם יהיו ההגדרות הקשורות לחיבור הלקוח והשרת, הגדרות שצריכות להיות זהות לזה של השרת.

קובץ Encryption_And_Decryption.py – קובץ בו 2 פונקציות הקשורות לאבטחת המידע של הלקוח: הצפנה ופיענוח. גם קובץ זה צריך להיות זהה בצד השרת, כדי שיצפינו ויפענו באופן זהה.

קובץ Handle_Images.py – קובץ האחראי על טיפול הצגת התמונות במסכי הGUI השונים באפליקציה.

קובץ Detection_app.exe קובץ אשר יריץ, ע"י הפעלתו בלחיצה, את הקובץ Client_Functions.py אשר יפעיל את שאר הקבצים לפי הקוד הרשום בו.

לאחר הורדת כל הקבצים הנ"ל יש ללחוץ על הקובץ Destection_app.exe ובכך יורץ הלקוח ויחד אתו האפליקציה.

* לאורך כל התהליך באפליקציה בהמשך יש למשתמש אפשרות נוספת שהיא לסגור את חלון האפליקציה, סגירת החלון תוביל ליציאה לא מסודרת מהאפליקציה ותסגור את החלון והתוכנה. ביציאה זו המשתמש עלול לאבד מידע.

בפני המשתמש יופיע מסך הפתיחה הבא:



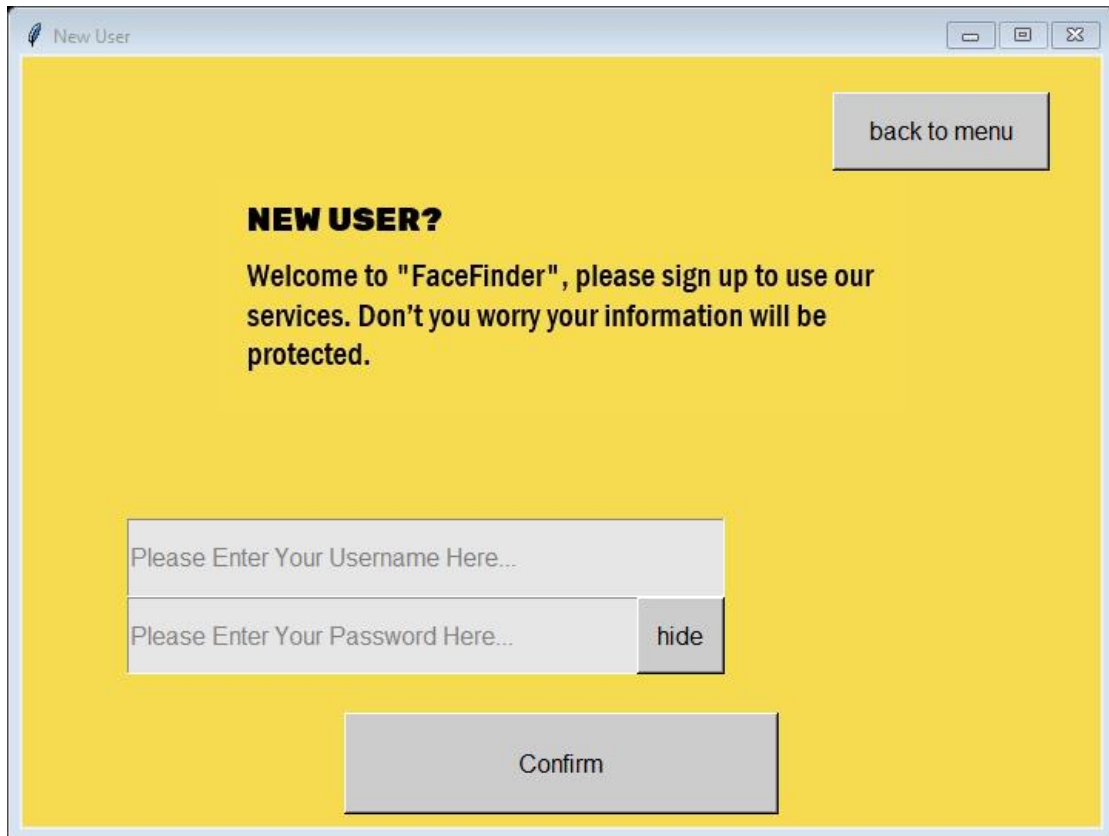
אלמוג גל. Face Founder.

ניתן לראות בתמונה את מסך הפתיחה, עם זכויות יוצרים על ידי, אלמוג גל, והלוגו המעוצב של האפליקציה. בפני המשתמש שתי אפשרויות:

כפתור login יוביל את המשתמש למסך בו יצטרך להכניס פרטי כניסה למשתמש קיים.

כפתור new user יוביל את המשתמש למסך בו יצטרך להכניס פרטי כניסה, שאותם יצטרך לזכור, ולפתוח משתמש חדש באפליקציה.

נניח שהמשתמש פעם ראשונה משתמש באפליקציה, ועל כן ילחץ על כפתור new user.



ניתן לראות את מסך יצירת משתמש חדש באפליקציה. בפני המשתמש מספר אפשרויות.

לחיצה על כפתור back to menu תחזיר את המשתמש למסך הפתיחה (ראה עמוד 24).

על המשתמש להכניס פרטים תקינים, שיהוו את מפתח הכניסה שלו למשתמש אישי.

נניח שהמשתמש לא מילא אף תא. מכיוון שהמשתמש לא מילא שם תא שם המשתמש האפליקציה דואגת להזכיר לו זאת.

New User

back to menu

NEW USER?

Welcome to "FaceFinder", please sign up to use our services. Don't you worry your information will be protected.

Please Enter Your Username Here...

Please Enter Your Password Here... hide

Confirm

Don't forget to enter a username.

נניח שהמשתמש כן מילא שם משתמש אך לא סיסמא.

New User

back to menu

NEW USER?

Welcome to "FaceFinder", please sign up to use our services. Don't you worry your information will be protected.

almog2

Please Enter Your Password Here... hide

Confirm

Don't forget to enter a password.

נניח שהמשתמש מילא שם משתמש קצר מידי.

New User

back to menu

NEW USER?

Welcome to "FaceFinder", please sign up to use our services. Don't you worry your information will be protected.

al

secret hide

Username must be at least 4 characters.

Confirm

נניח שהמשתמש מילא סיסמא קצרה מידי.

New User

back to menu

NEW USER?

Welcome to "FaceFinder", please sign up to use our services. Don't you worry your information will be protected.

almog2

sec hide

Password must be at least 4 characters.

Confirm

נניח שהמשתמש מילא שם משתמש שכבר קיים במערכת, ונמצא בשימוש.

New User

back to menu

NEW USER?

Welcome to "FaceFinder", please sign up to use our services. Don't you worry your information will be protected.

almog

secret

hide

Username is already being use, please use another.

Confirm

נניח שהמשתמש נמצא במקום ציבורי ולא מעוניין לראות את סיסמתו, לכן לוחץ על כפתור hidden.

New User

back to menu

NEW USER?

Welcome to "FaceFinder", please sign up to use our services. Don't you worry your information will be protected.

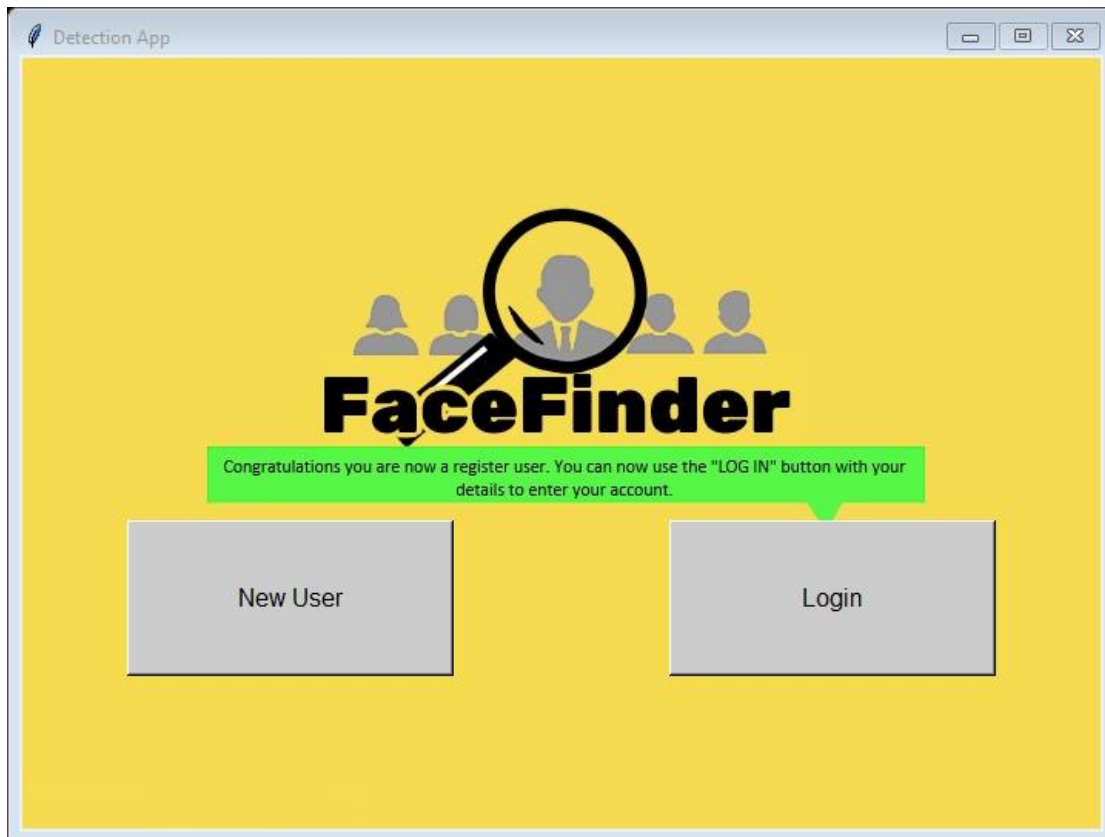
almog

hide

Confirm

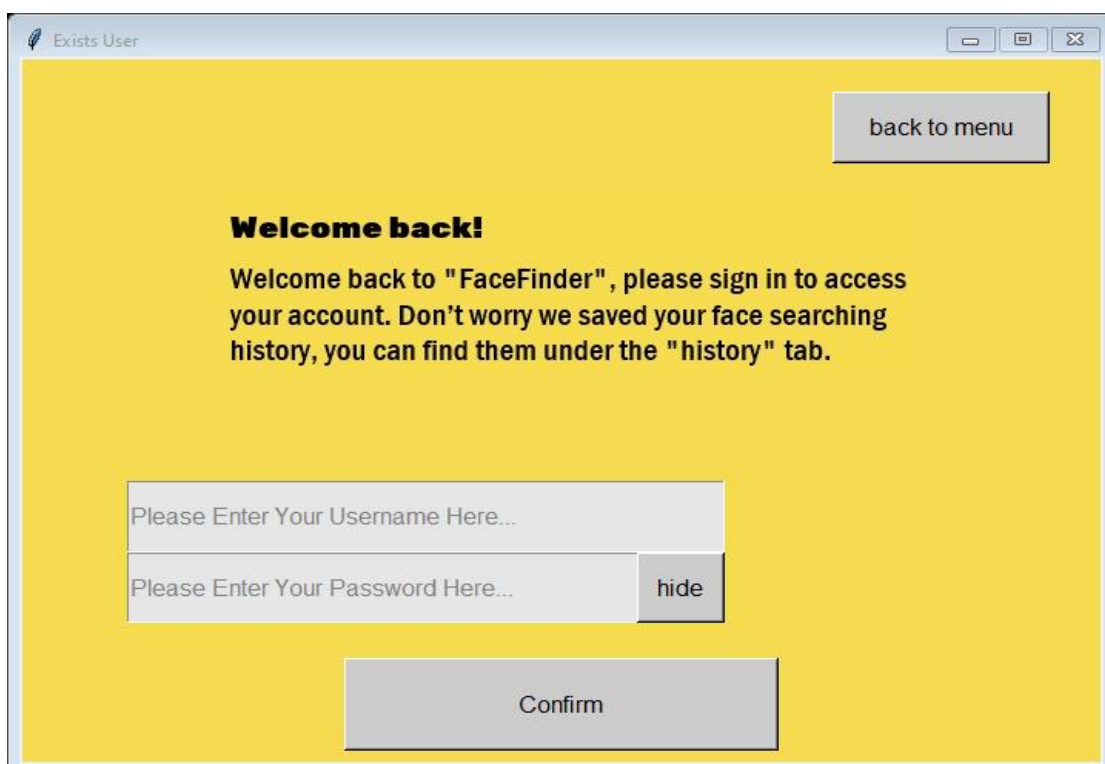
אלמוג גל. Face Founder.

לאחר הכנסת שם משתמש וסיסמא תקינים ולחיצה על כפתור confirm יחזור המשתמש למסך הפתיחה ויקבל הודעה שהרשמתו בוצעה בהצלחה. ועליו להיכנס לחשבון שלו עם הפרטים שלו כדי להשתמש באפליקציה.



שוב, חזרה למסך הפתיחה (ראה עמוד 24).

כעת המשתמש רוצה להיכנס לחשבון שלו. לכן הוא לוחץ על כפתור login.



בפני המשתמש אפשרויות.

לחיצה על כפתור ה`back to menu` תחזיר אותו למסך הפתיחה (ראה עמוד 24).

נניח שהמשתמש הכניס פרטים שגויים – שם המשתמש לא קיים במערכת או הסיסמא לא קשורה לשם המשתמש.

The screenshot shows a web browser window titled "Exists User". The page has a yellow background. At the top right is a "back to menu" button. In the center, it says "Welcome back!" followed by a message: "Welcome back to 'FaceFinder', please sign in to access your account. Don't worry we saved your face searching history, you can find them under the 'history' tab." Below this are two input fields: "Please Enter Your Username Here..." and "Please Enter Your Password Here...". The password field has a "hide" button next to it. At the bottom center is a "Confirm" button. On the right side, a red error message box says: "Your username or password are incorrect. If you are new please sign up first (under the tab 'New User')." The input fields are currently empty.

This screenshot is identical to the one above, but the input fields now contain the text "almog2" for the username and "hacker" for the password. The error message box remains visible on the right.

נניח שהמשתמש נמצא במקום ציבורי ולא מעוניין שיראו את סיסמתו.

Exists User

back to menu

Welcome back!

Welcome back to "FaceFinder", please sign in to access your account. Don't worry we saved your face searching history, you can find them under the "history" tab.

almog2

**** hide

Confirm

המשתמש מכניס כעת פרטים תקינים ונכונים ולחץ על confirm ועובר למסך הבא. המסך הבא הינו מסך הוראות שנועד להסביר למשתמש כיצד להשתמש באפליקציה. (הערת כותב: החלון לא עדכני, ההוראות שונו ומסך יעוצב בהתאם ברגע שההוראות יהיו סופיות).

How To Use

HOW TO USE

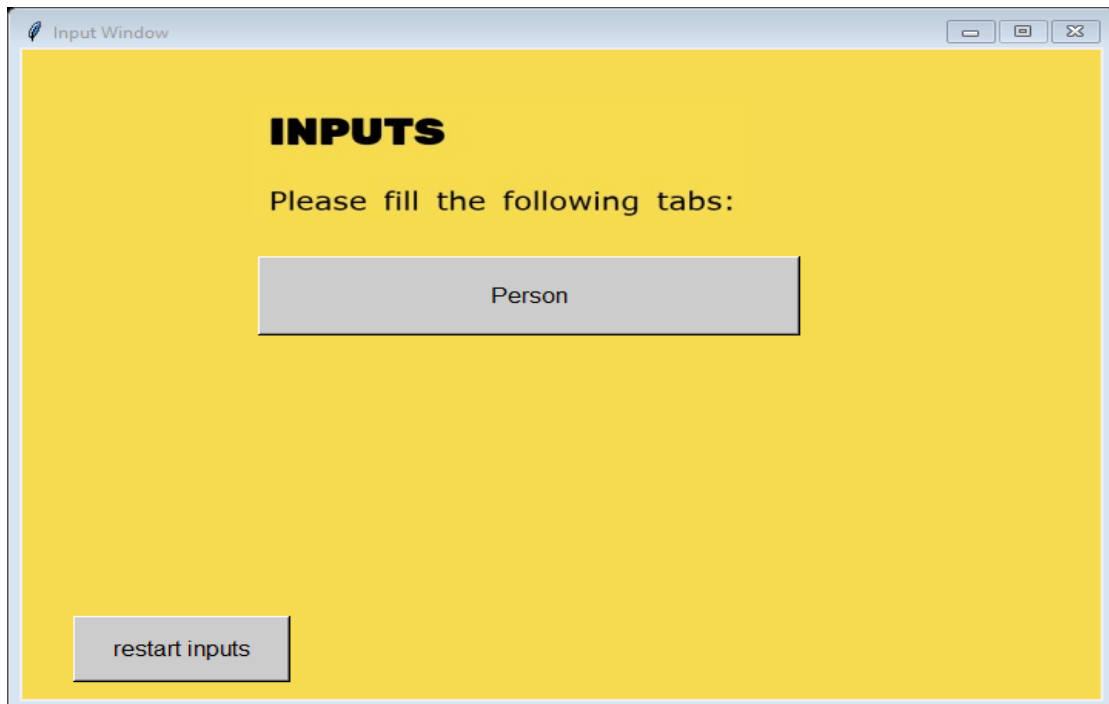
1. Upload someone's face in a picture (a .jpg or a .png) by inputting the picture's path. Please make sure the picture is clear and bright as best you can, to improve your final outputs. Note that if you upload more than one person, the app will look for pictures that contain all people together.
2. Input a path of a gallery which you want to find the person in. Pay attention the files that are not a picture type (a .jpg or a .png) will not be check.
3. The App will notify you about the pictures that the app can't fully decide if the person in the picture is the desired person. You can decide if the person in those picture is a match or not. The app will automatically presume that it is a match if you skip this step. So, to accuracy your results don't skip this part.
4. If there are pictures which the person is unclear, the chances of founding the desire person will be reduce.
5. At the end of the procedure, the pictures that found out to be a match with the will be automatically save in your "History" tab. To access to your history you will have first to log into your account.

Also you will have the option to save your match pictures on your PC.

next

אלמוג גל. Face Founder.

לאחר עיון רב בהוראות המשתמש הפנים כיצד להשתמש באפליקציה, ומה השימוש של האפליקציה ולוחץ על כפתור next כדי להמשיך הלאה.

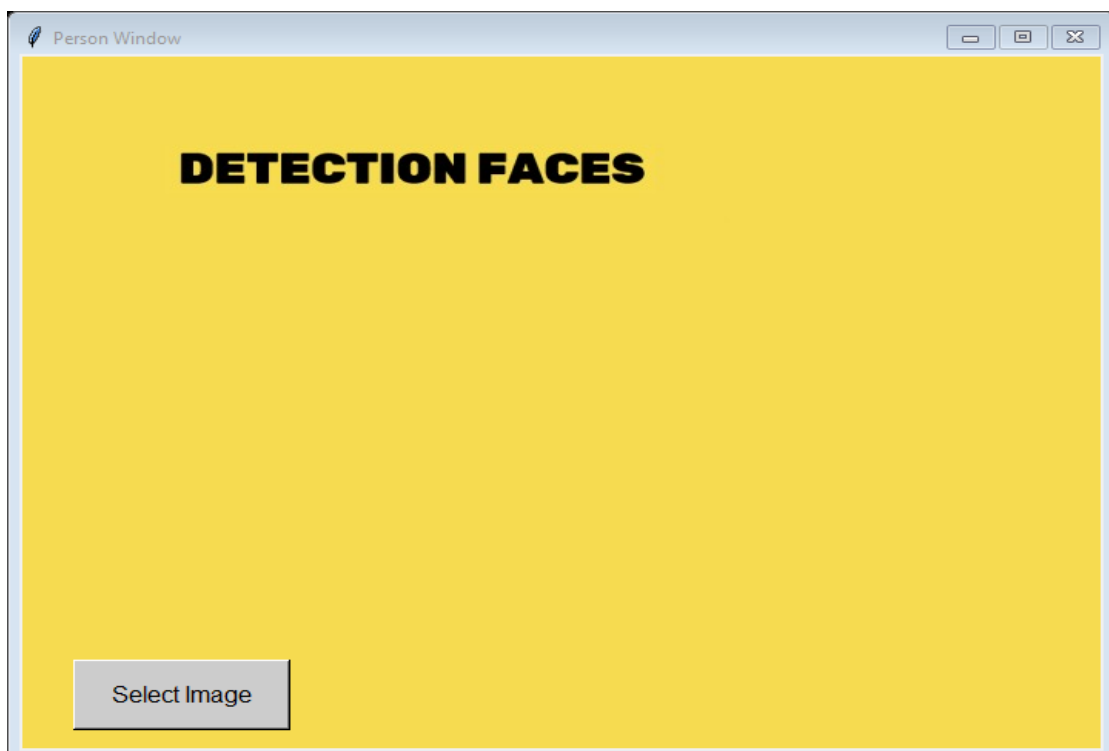


בפני המשתמש מסך קלט. לפני המשתמש כמה אפשרויות:

לחיצה על כפתור back שתחזיר את המשתמש להוראות האפליקציה (כפתור זה עדיין לא עוצב + ראה עמוד 31).

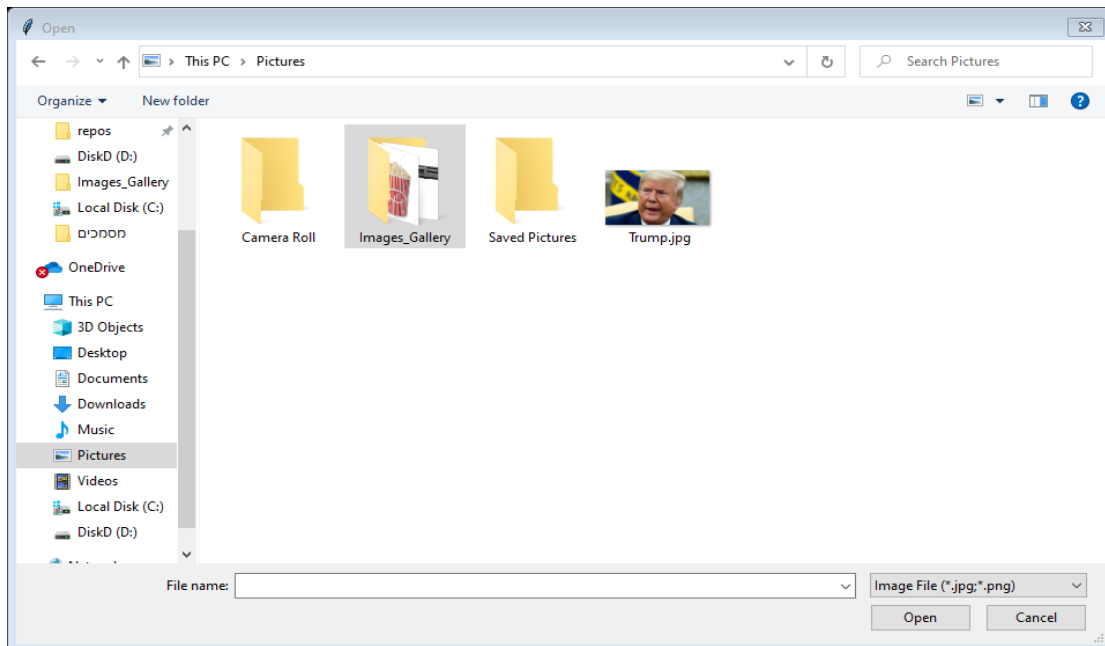
לחיצה על כפתור restart inputs בכל שלב תאפס ותמחק את כל הקלט שהכניס עד כה המשתמש, ותחזיר אותו למסך הקלט (ראה עמוד 32).

על מנת להשתמש ולחפש אדם מסוים מתוך גלריית תמונות המשתמש לוחץ על כפתור Person.



אלמוג גל. Face Founder.

במסך "זיהוי פנים" זה המשתמש ילחץ על כפתור select image כדי לבחור תמונה של האדם אותו רוצה לחפש.



לחיצה על הכפתור פתחה דיאלוג של המשתמש עם File Explorer ודיאלוג של הלקוח עם ה File Explorer. על הלקוח לבחור תמונה בה מופיע האדם שהוא מחפש. בנוסף, הדיאלוג מגביל את המשתמש לבחור רק קבצי jpg. או png. ועל כן, הוא לא יוכל לבחור קובץ מסוג לא נכון.

לחיצה על כפתור האינסוסטית ודיאלוג תוביל את הלקוח למסך "זיהוי פנים" (ראה עמוד 32).

לחיצה על תמונה בה יש יותר מפנים אחד תראה למשתמש חלון עם הערה שהוא הכניס תמונה עם יותר מפנים אחד (התמונה עדיין לא עוצבה).

לחיצה על תמונה בה יש פחות מפנים אחד תראה למשתמש חלון עם הערה שהוא הכניס תמונה עם פחות מפנים אחד (התמונה עדיין לא עוצבה).

המשתמש בוחר תמונה אותה הוא רוצה לחפש (נניח, Trump.jpg).

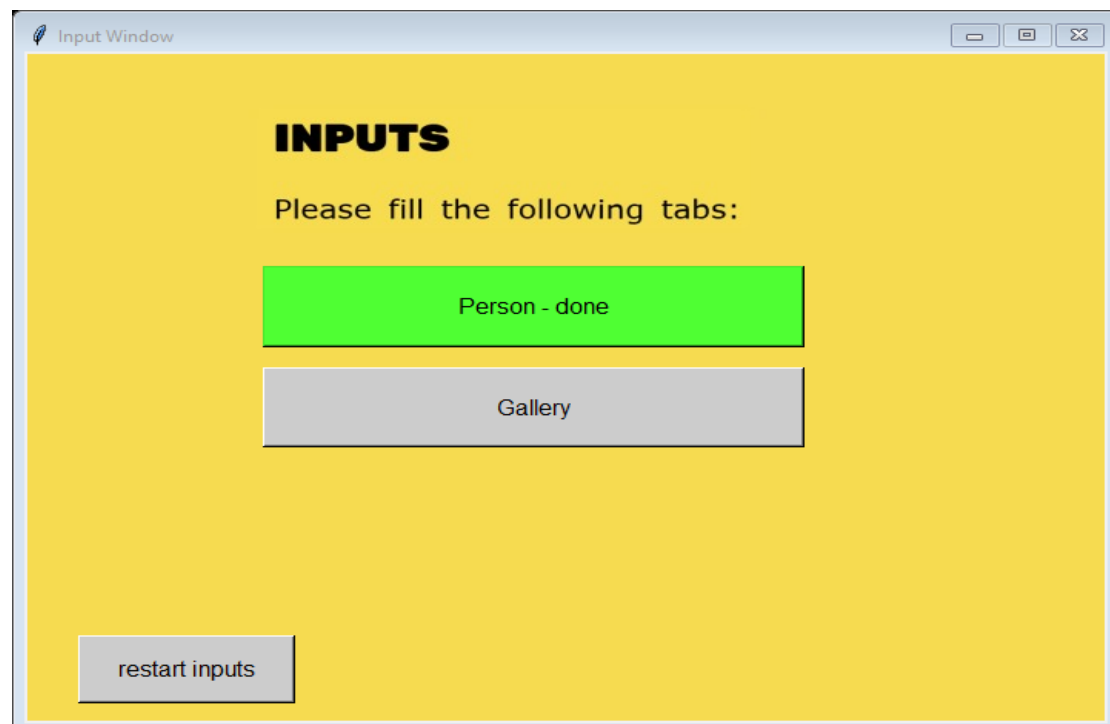


אלמוג גל. Face Founder.

ללקוח יפתח החלון בו יוכל לראות בבירור איזה בן אדם הוא בחר. בפני המשתמש שתי אפשרויות:

ללחוץ על כפתור change image שתוביל אותו לדיאלוג חוזר על File Explorer (ראה עמוד 33).

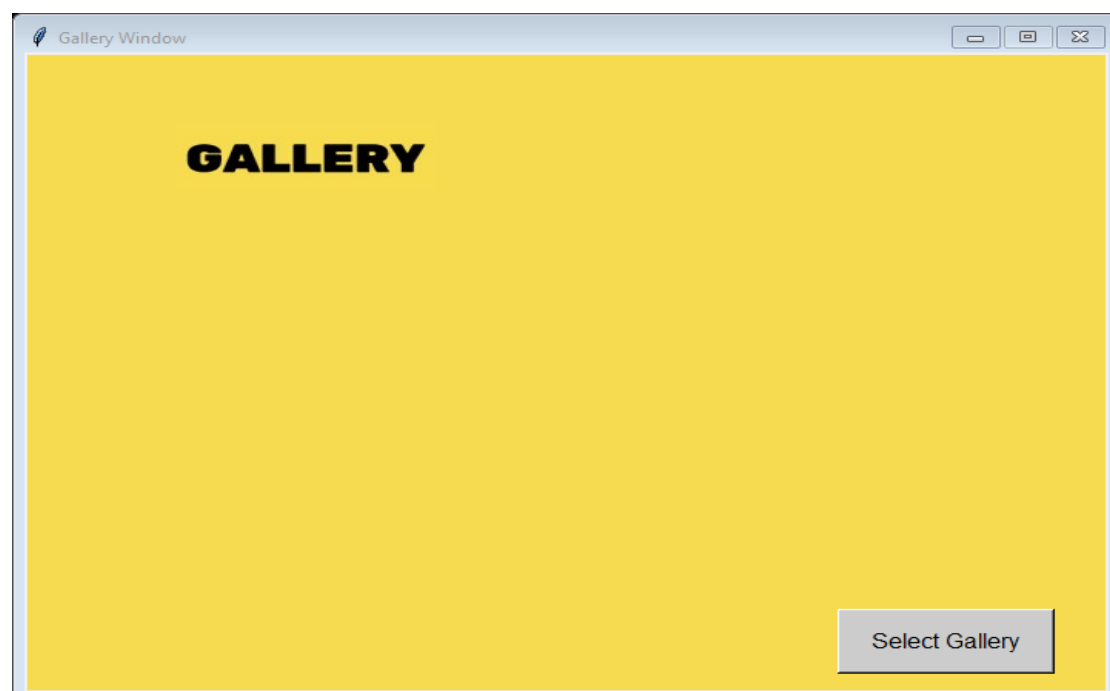
ללחוץ כפתור next מה שאומר שזה האדם שברצונו לחפש.



לאחר שהמשתמש בחר אדם, האפליקציה מודיעה לו שאת החלק הזה הוא סיים ועליו עכשיו לבחור גלריה.

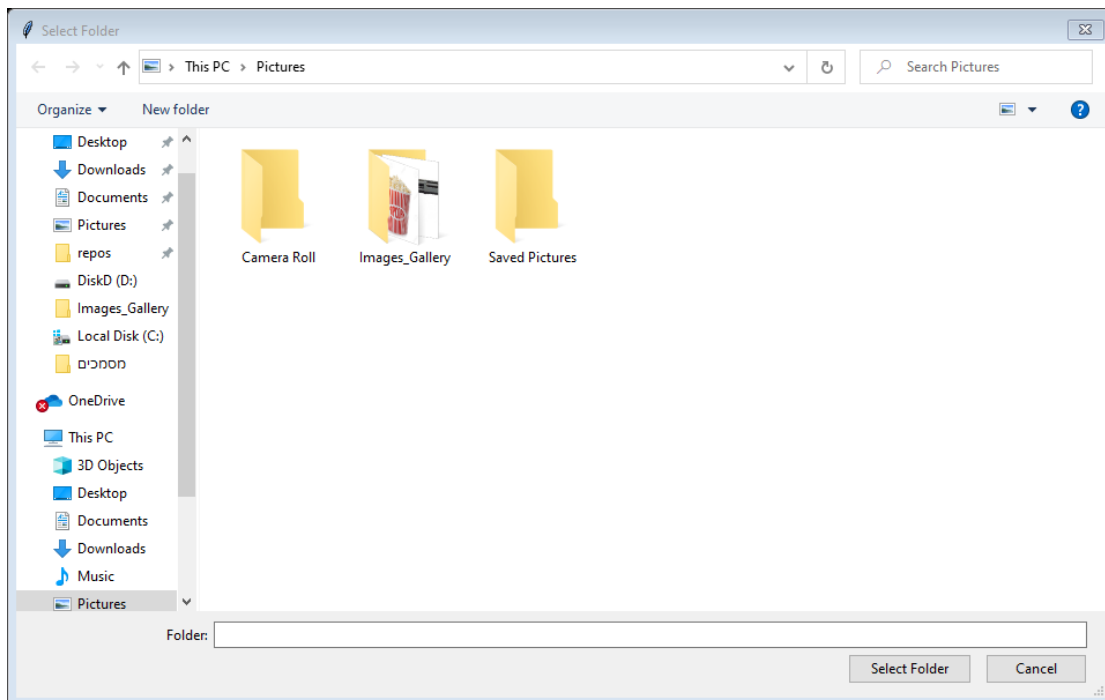
לחיצה כל כפתור restart inputs בכל שלב תאפס ותמחק את כל הקלט שהכניס עד כה המשתמש, ותחזיר אותו למסך הקלט (ראה עמוד 32).

המשתמש רוצה להמשיך בתהליך ולכן לוחץ על כפתור gallery.



אלמוג גל. Face Founder.

נפתח חלון "גלריה", בפני המשתמש כפתור select gallery כדי להתחיל לבחור גלריה.

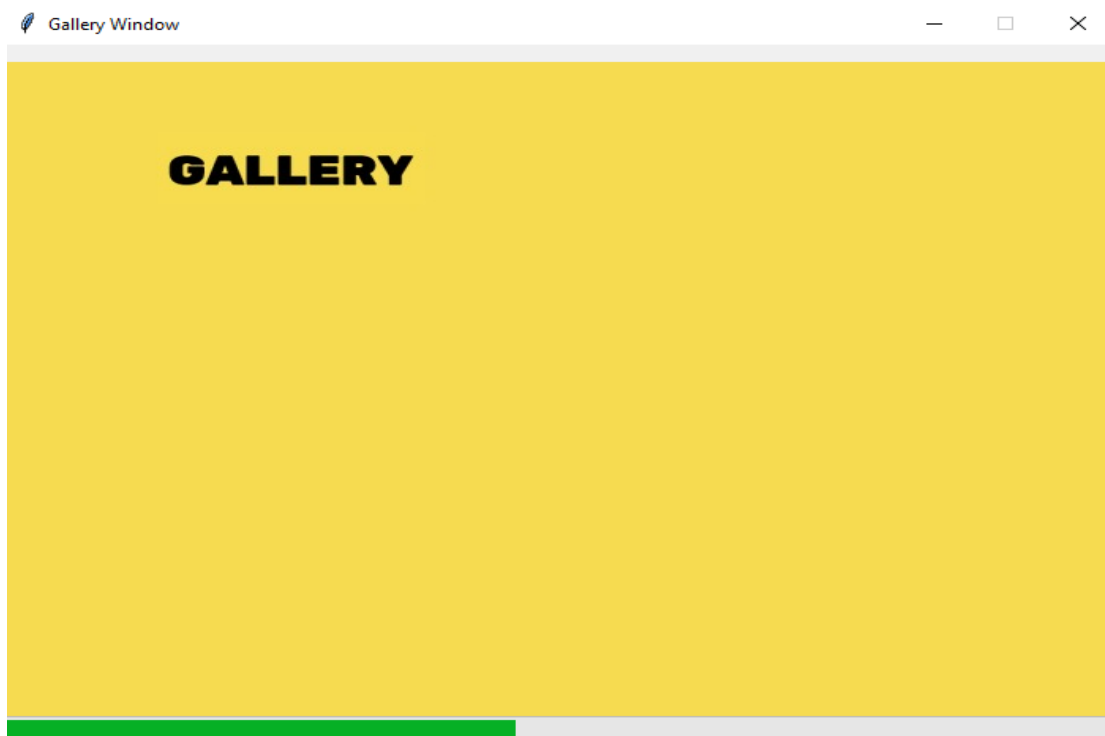


לחיצה על הכפתור פתחה דיאלוג של המשתמש עם File Explorer ודיאלוג של הלקוח עם ה File Explorer. על הלקוח לבחור תמונה בה מופיע האדם שהוא מחפש. בנוסף, הדיאלוג מגביל את המשתמש לבחור רק תיקיות. ועל כן, הוא לא יוכל לבחור קבצים למיניהם.

לחיצה על כפתור האינסוסט וסגירת הדיאלוג תוביל את הלקוח למסך "גלריה" (ראה עמוד 34).

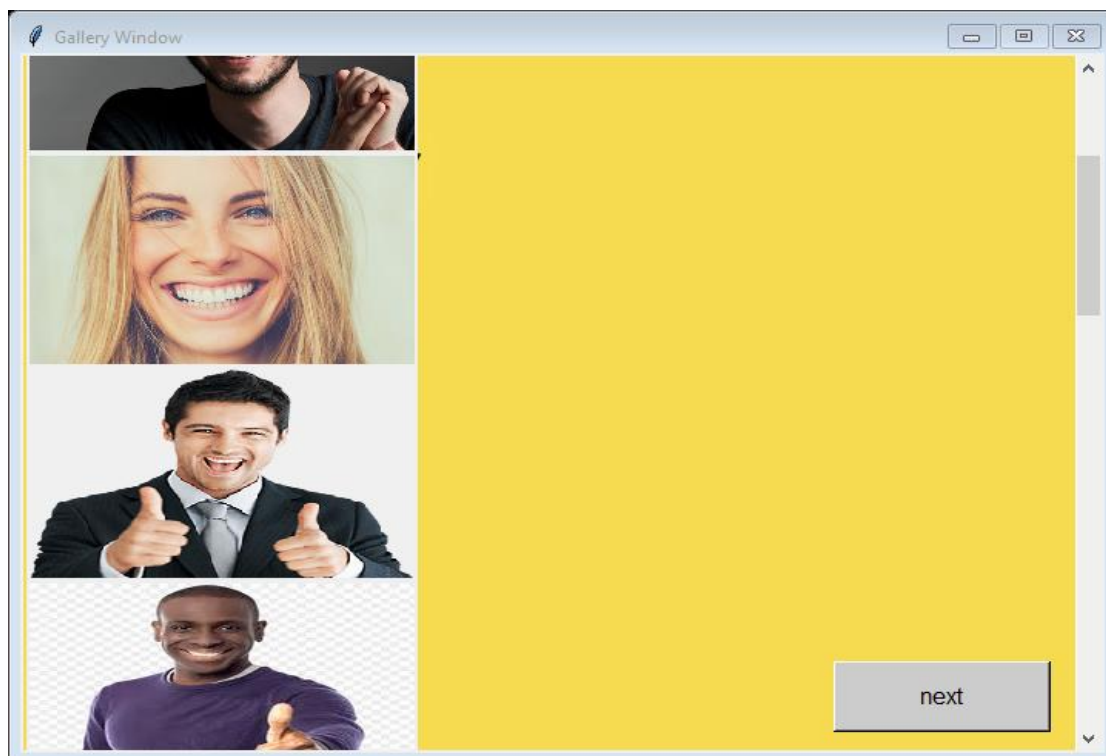
המשתמש מעוניין לבחור בגלריית תמונות שנמצאת לו על המחשב ולכן הוא בוחר אותה (נניח images_gallery).

לפני המשתמש נפתח מסך טעינה, שמראה לו עוד כמה בערך זמן נשאר להעלאת התמונות.

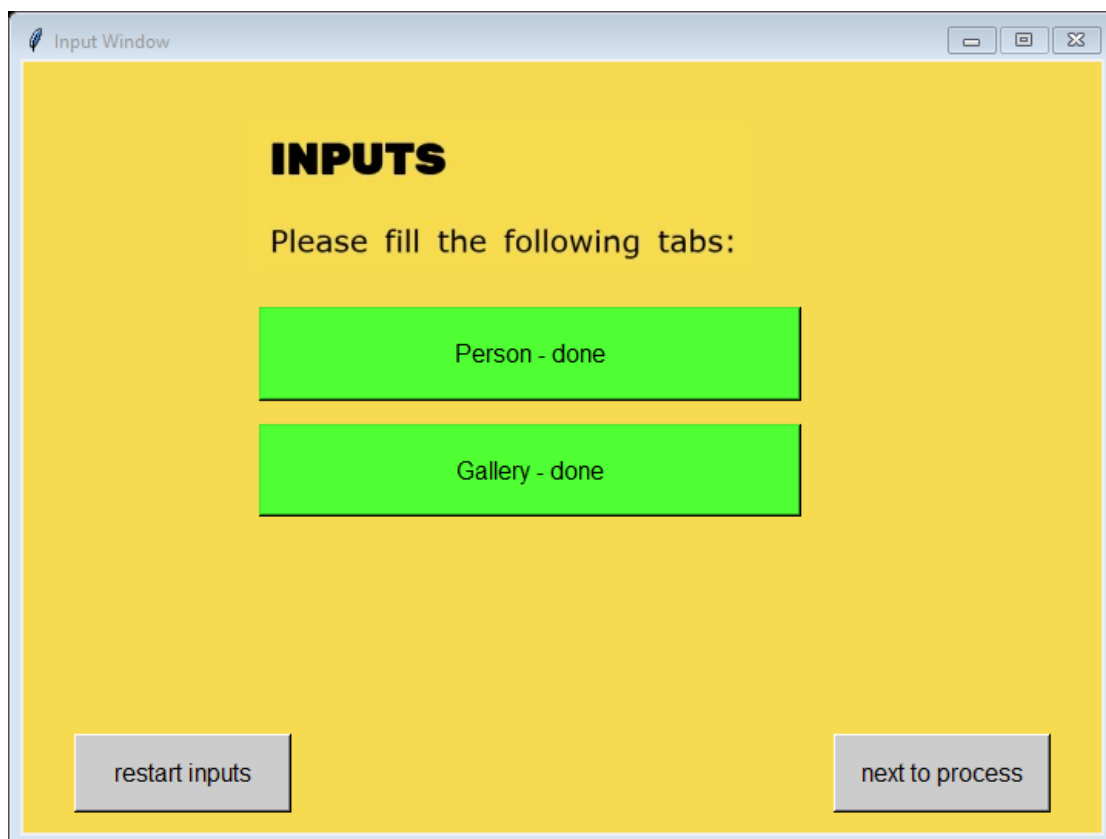


אלמוג גל. Face Founder.

בסיום הטעינה יופיע למשתמש המסך הבא.



לפני המשתמש נפתח מסך, עם scroll bar, בה יוכל לעיין בכל התמונות בגלריה שהעלה. בנוסף לרשותו כפתור next יעביר אותו לחלון סיום הקלט.



אלמוג גל. Face Founder.

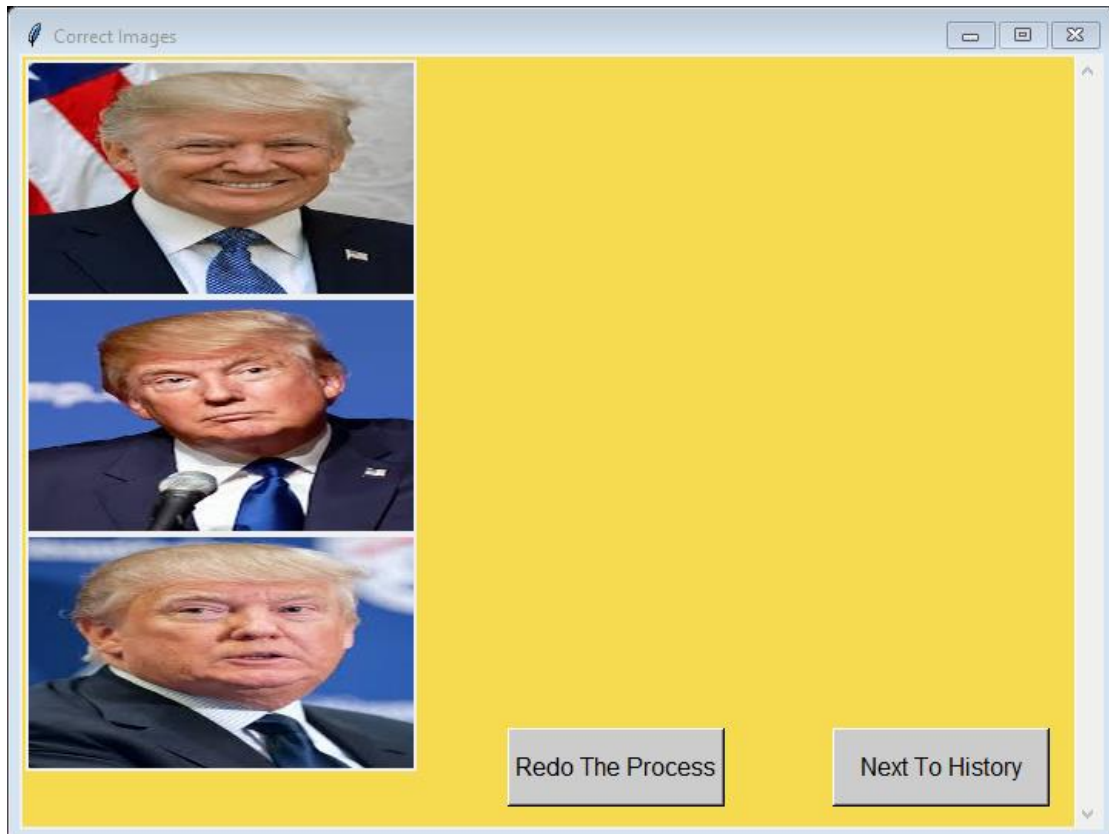
המשתמש סיים וידוע שכל הקלט עבר בהצלחה.

לחיצה כל כפתור restart inputs בכל שלב תאפס ותמחק את כל הקלט שהכניס עד כה המשתמש, ותחזיר אותו לסמך הקלט (ראה עמוד 32).

המשתמש מעוניין להמשיך לתהליך כדי לדעת באילו תמונות שהעלה האדם אותו חיפש נמצא (בניח דונאלד טראמפ).

בפני המשתמש נפתח מסך טעינה, עד שכל התמונות יעלו.

לאחר סיום הטעינה יפתח מסך ה"תמונות הנכונות".

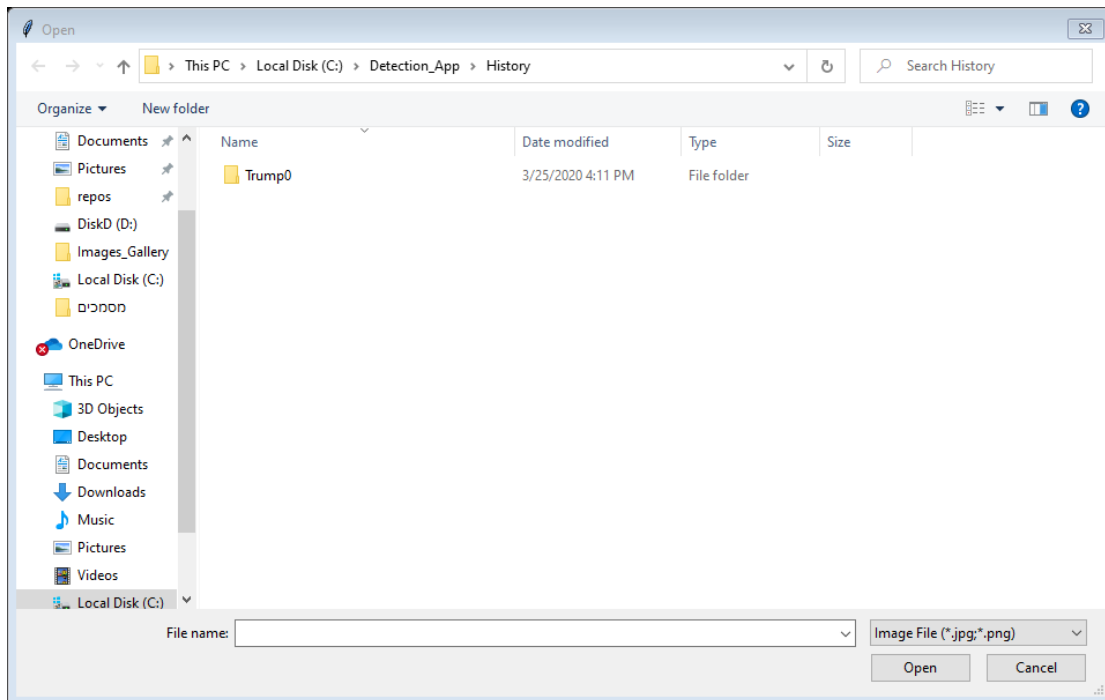


במסך זה למשתמש 2 אפשרויות:

ללחוץ על כפתור redo the process, לחיצה על כפתור זה תוביל את המשתמש למסך הקלט הראשוני (ראה עמוד 32), שם יתחיל את התהליך מחדש. המידע והתהליך הנוכחי ישמר לו אצל השרת אבל עוד לא יעלה להיסטוריית הלקוח.

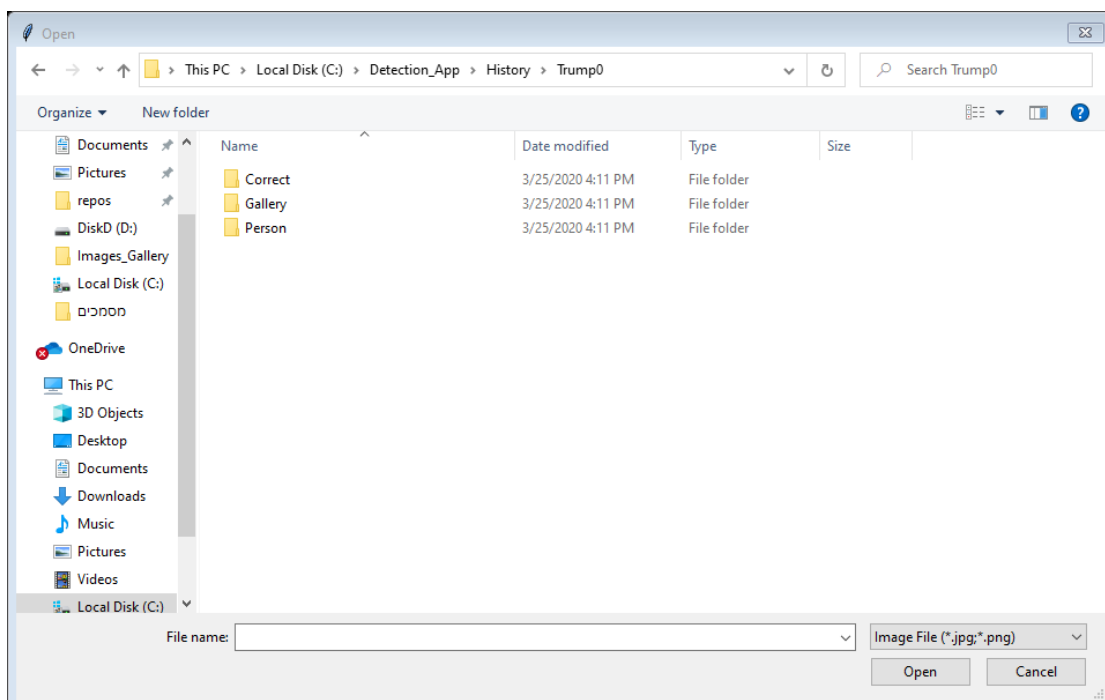
לחיצה על כפתור next to history כשאר המשתמש עשה מספר מסוים של תהליכים וברצונו להסתכל על היסטוריית התהליכים האישית שלו. יפתח מסך טעינה אשר יטען עד סיום העלאת קבצי ההיסטוריה החדשים ללקוח.

אלמוג גל. Face Founder.



בסיום הטעינה נפתח דיאלוג משולש עם File Explorer. בתיקייה בה ישנם כל החומרים של האפליקציות השונות במחשב ה-windows (כרגע תיקייה סתמית). שם יראה הלקוח את כל האנשים אשר חיפש. המשתמש ביצע רק תהליך אחד, וחיפש בפעם הראשונה את דונאלד טראמפ, ועל כן ישנה רק תיקייה אחת הנקראת Trump0, כי זהו החיפוש הראשון בוצע על דונאלד טראמפ.

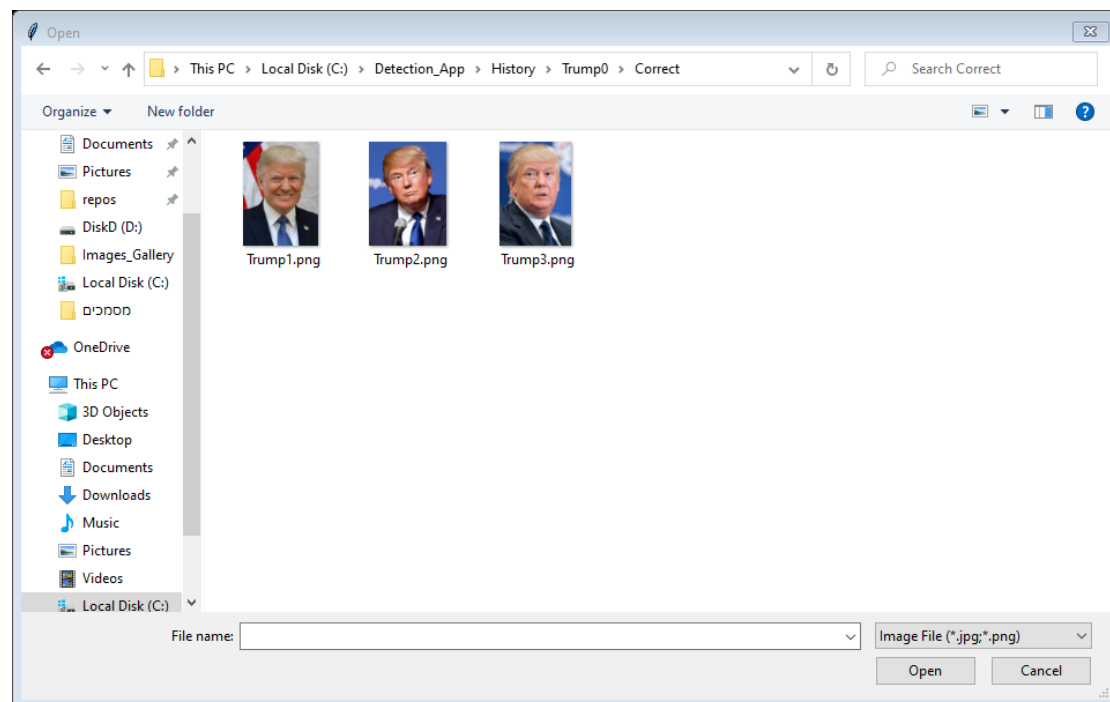
נניח שהמשתמש רוצה עוד מידע על התהליך מסוים שביצע (נניח התהליך הראשון שביצע על דונאלד טראמפ). המשתמש ילחץ על התיקייה Trump0. לחיצה על כפתור החזרה במעלה העמוד תוביל את המשתמש לpath הקודם שעיון בו. כך יוכל המשתמש לעיון ברחבי בגלריה שלו בחופשיות דרך האפליקציה.



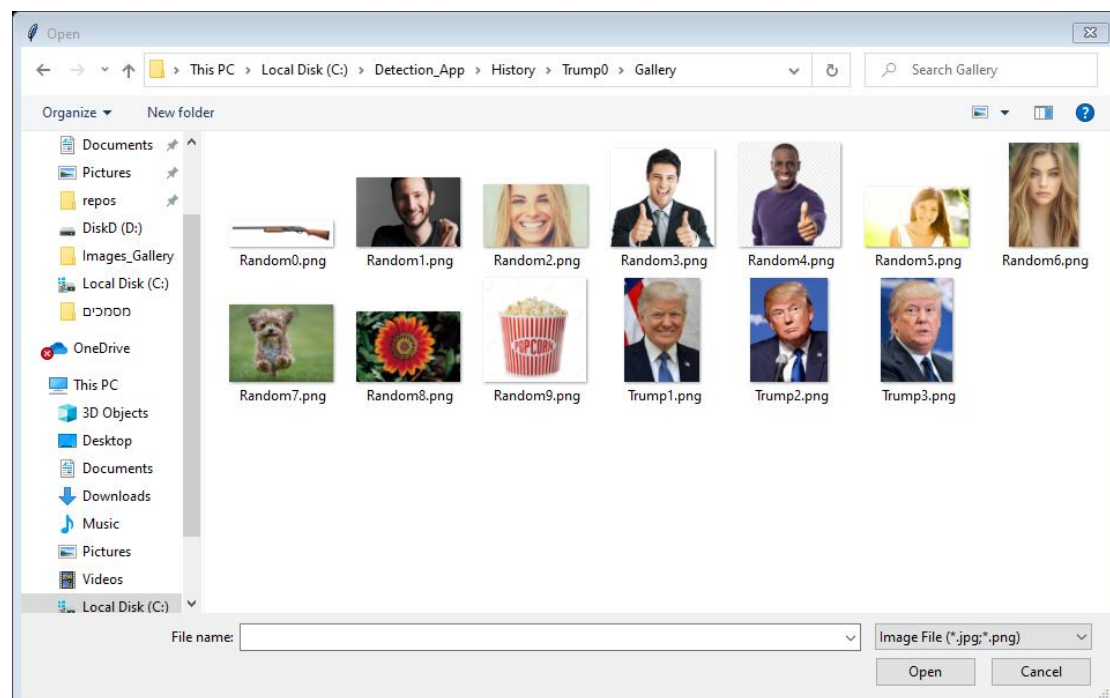
אלמוג גל. Face Founder.

בפני המשתמש מידע על התהליך שביצע ב-3 תיקיות מסודרות.

Correct – התמונות הנכונות, אלו שנמצא בהם האדם המבוקש (בניח דונאלד טראמפ).

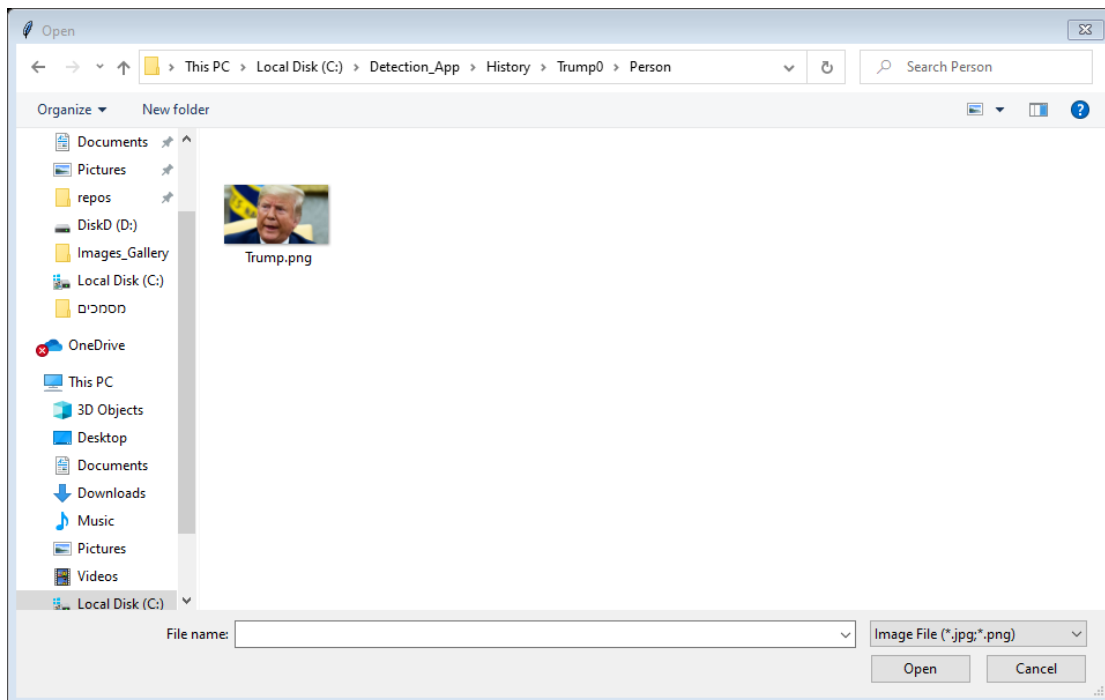


Gallery – גלריית התמונות בהם המשתמש חיפש את האדם המבוקש (בניח דונאלד טראמפ).

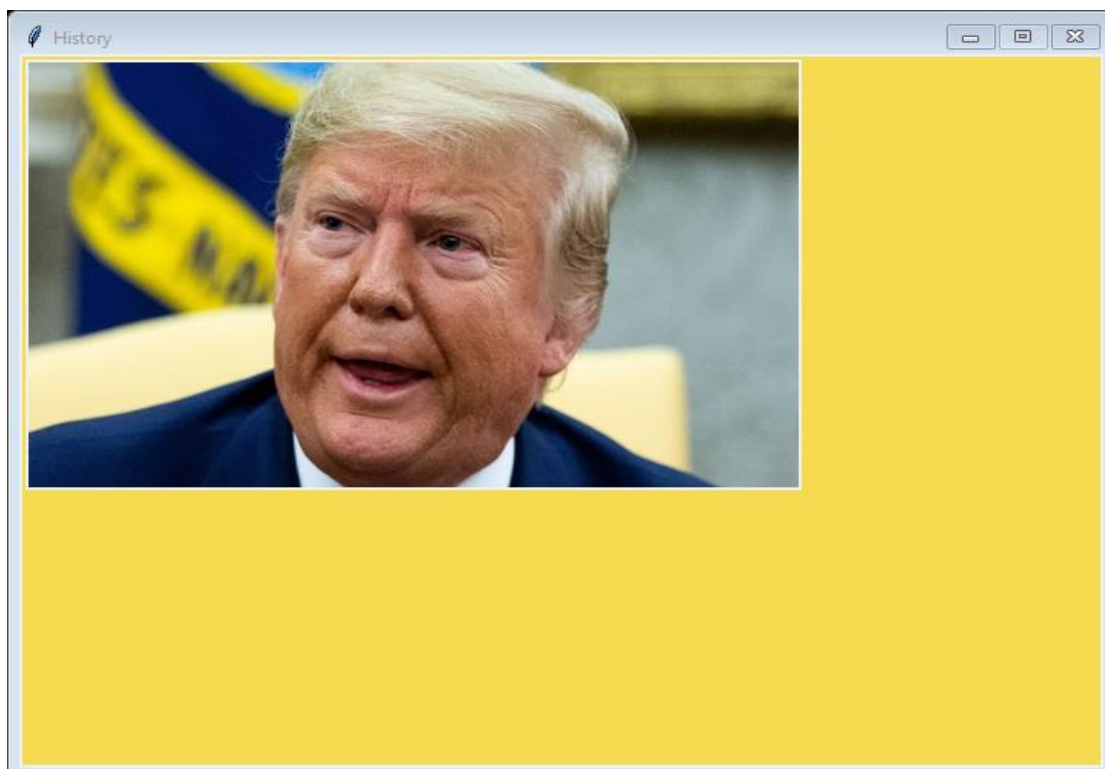


אלמוג גל. Face Founder.

Person – האדם אותו המשתמש רצה לחפש (נניח דונאלד טראמפ).



המשתמש רשאי או ללחוץ על כפתור האיקס שיוביל אותו לחלון היסטוריה ריק, או ללחוץ על תמונה שתוביל אותו לחלון היסטוריה שבו התמונה שבחר מוצגת. נניח שהמשתמש רוצה תמונה מסוימת (נניח דונאלד טראמפ). הוא לוחץ עליה.



בתחתית העמוד יהיה למשתמש 2 אפשרויות לעשות את התהליך שוב שיחזיר אותו לעמוד קלט ריק (ראה עמוד 32), או סיום התהליך ויציאה מהאפליקציה שיסגור את התוכנה ואת החלון למשתמש ותוביל ליציאה מסודרת. הכפתורים עדיין לא עוצבו.

מדריך למפתח

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Client_Functions.py	General Client	כל הפונקציות המרכזיות של הלקוח. הפעלה וסנכרון פונקציות אלו על פי מעשי המשתמש, תוך כדי הצגת ממשק גרפי. שליחת הודעות לשרת בהתאם. יצירת מפתח סימטרי פרטי עם השרת.	לדאוג באופן כללי, שהלקוח פועל באופן תקין, נוחות המשתמש ותקשורת עם רציפה ותיקינה עם השרת.	Client\ Client_Functions.py

תדפיס הקוד:

הכנה של משתנים גלובליים: paths של רקעים, גודל ראשוני של המסך, ערכים ראשוניים למשתנים למיניהם בתוכנית, סוגי הקבצים שהמשתמש רשאי לעלות, events, רקעים, paths של תיקיות היסטוריה ותיקיות זמניות במחשב הלקוח.

משתנים קבועים: בכל הפונקציות קיים משתנה username – נועד כדי להזדהה בשליחת ההודעות לשרת.

בקשה לקבלת מפתח ציבורי מהשרת.

בקשה ליצירת מפתח סימטרי פרטי מהשרת, יצירת מפתח סימטרי פרטי ושמידתו אצל הלקוח.

מחיקת כל הווידג'טים הקודמים, פתיחת החלון הפתיחה, יצירת רקע, שני כפתורים אפורים: כניסה למערכת- יוביל להכנסת פרטי חשבון קיים ומשתמש חדש- הכנסת פרטים של חשבון חדש, כותרת, event של יציאה של המשתמש באופן לא מסודר. **משתנים:** background – path של רקע המסך בהתאם למצב שבו נמצא המשתמש.

מחיקת כל הווידג'טים הקודמים, החלפת החלון של משתמש חדש. יצירת רקע, כותרת – "חשבון חדש", שני תאי מילוי אפורים: אחד רשום בתוכו "פה הכנס שם משתמש" שנמחק ברגע שהמשתמש נמצא על התא ושם המשתמש יכניס את שם המשתמש ושני רשום בתוכו "פה הכנס את הסיסמא" שנמחק ברגע שהמשתמש לו נמצא על התא. שני events אחד של חץ למעלה ואחד של חץ למטה שמאפשרים לנוע, בנוסף על ללחיצות, בין התאים. Event נוסף של לחצן enter שיפעיל את כפתור האישור. כפתור "חזור" שיחזיר את המשתמש למסך הפתיחה. כפתור "אישור" שישלח אל השרת את הפרטים של ניסיון ההתחברות לחשבון חדש. במידה וישנה איזושהי תקלה השרת ישלח ללקוח הודעת שגיאה ובה ירשום את התקלה. השרת יקרא לפונקציה הזאת וישנה את הרקע לרקע שגיאה בהתאם לטעות. במקרה של הכנסת פרטים נכונה, הלקוח יחזור למסך הפתיחה שם ישונה הרקע לרקע שבו יש הודעה על הצלחת פתיחת החשבון החדש. כפתור החבא אשר יסתיר את הסיסמא, יקרא לפונקציה החבא סיסמא. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** background – path של רקע המסך בהתאם למצב שבו נמצא המשתמש, username entry

– מה היה בתא המילוי של שם המשתמש, במידה ויש קריאה חוזרת לפונקציה, entry –password – מה היה בתא המילוי של שם הסיסמא, במידה ויש קריאה חוזרת לפונקציה.

מחיקת כל הווידג'טים הקודמים, החלפת חלון כניסה לחשבון קיים. יצירת רקע, כותרת – "כניסה", שני תאי מילוי אפורים: אחד רשום בתוכו "פה הכנס שם משתמש" שנמחק ברגע שהמשתמש נמצא על התא ושם המשתמש יכניס את שם המשתמש ושני רשום בתוכו "פה הכנס את הסיסמא" שנמחק ברגע שהמשתמש לו נמצא על התא. שני events אחד של חץ למעלה ואחד של חץ למטה שמאפשרים לנוע, בנוסף על ללחיצות, בין התאים. Event נוסף של לחצן enter שיפעיל את כפתור האישור. כפתור "חזור" שיחזיר את המשתמש למסך הפתיחה. כפתור "אישור" שישלח אל השרת את הפרטים של ניסיון ההתחברות לחשבון קיים. במידה וישנה איזושהי תקלה השרת ישלח ללקוח הודעת שגיאה ובה ירשום את התקלה. השרת יקרא לפונקציה הזאתי וישנה את הרקע לרקע שגיאה בהתאם לטעות. במקרה של הכנסת פרטים נכונה, הלקוח יעבור לחלון הוראות, וכניסתו לחשבון בוצעה בהצלחה. כפתור החבא אשר יסתיר את הסיסמא, יקרא לפונקציה החבא סיסמא. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש, username entry – מה היה בתא המילוי של שם המשתמש, במידה ויש קריאה חוזרת לפונקציה, password entry – מה היה בתא המילוי של שם הסיסמא, במידה ויש קריאה חוזרת לפונקציה.

מחיקת כל הווידג'טים הקודמים, החלפת רקע וחלון לרקע וחלון הוראות, כותרת – "איך משתמשים", event של לחיצה על enter לעבור לחלון הקלט הראשוני, ויצירת כפתור הבא של לחיצה עליו גם תעבור לחלון הקלט הראשוני. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש.

מחיקת כל הווידג'טים הקודמים, החלפת חלון לחלון הקלט, החלפת רקע לרקע הקלט, כותרת – "חלון קלט", כפתור איפוס קלט אשר מאפס את כל הקלטים שהוכנסו בסשן הזה וקורא לפונקציה מחדש (בנוסף, השרת מוחק תהליכים שלא הושלמו ולכן קיימת פה מחיקת תהליך). איפוס event על לחיצת enter ללא לעשות כלום. אם עוד לא הוכנס קלט של אדם יוצג כפתור שרשום עליו אדם, שלחיצה עליו תעבור למסך קליטת אדם. אם הוכנס קלט של אדם, אז לא הוכנס קלט של גלריה ולכן תוצג הודעה שהאדם נקלט בהצלחה, יחד עם כפתור הכנס גלריה שלחיצה עליו תעבור לחלון קלט הגלריה. אם הוכנס גלריה, סימן שגם הוכנס אדם, ולכן 2 הקלטים סוימו בהצלחה, יוצג הודעה שגם האדם וגם הגלריה נקלטו בהצלחה, ויוצג כפתור המשך לתהליך שלחיצה עליו תוביל לתהליך הפנים. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש, אדם- בוליאני, האם בוצע קלט של אדם או לא, גלריה- האם בוצע קלט של גלריה או לא (המעיד על סיום הקלט, קיים סדר כרונולוגי בין האדם לגלריה).

מחיקת כל הווידג'טים הקודמים, החלפת חלון לחלון הקלט של אדם, החלפת הרקע לרקע קלט של אדם, כותרת – "חלון אדם", יוצג כפתור של בחירת תמונה שלחיצה עליו תוביל לפונקציית דיאלוג בין המשתמש לסייר קבצים. אם קיים איזשהו path סימן שהמשתמש כבר בחר תמונה תקינה, על כן הלקוח ישלח בקשה לשרת וישאל אותו האם בתמונה הזאת יש פרצוף אחד. אם אין פנים שווה 2 סימן שיש יותר מפנים אחד בתמונה, על פי השרת, ויקראו לפונקציה הזאת שוב עם רקע שממנו יוצג טעות שלפיה יש יותר מפרצוף אחד. אם פנים שווה ל0 סימן שאין פנים בתמונה, על פי השרת, ויקראו לפונקציה הזאת שוב עם רקע שממנו יוצג טעות שלפיה יש אין פנים. אם פני שווה ל1 סימן שבתמונה יש אדם אחד, והיא תקינה לחלוטין. הלקוח יבחר את שם התיקיה בה ישמור הלקוח את הפרטים לתמונה הנ"ל לפי תא מילוי. לאחר המילוי ישלח שם התיקיה והמידע הבינארי של התמונה הנ"ל לשרת

בבקשה לפתוח תיקיית היסטוריה תחת השם הזה, לתמונה הנ"ל. השרת יאשר. הלקוח ירצה להציג למשתמש את התמונה אשר בחר. לכן יפתח path זמני שם ישמור את התמונה, ישנה את גודלה לגודל שיתאים למסך, ישמור אותה כpng (בtkinter ניתן להציג רק תמונות מסוג png), ויציא אותה. לאחר הצגת התמונה יופיע למשתמש שתי כפתורים: כפתור בחר תמונה אחרת שתקרא לפונקציה הנ"ל מהתחלה שוב (והשרת ימחק את התהליך הנ"ל, כי עוד לא הושלם) וכפתור הבא שלחיצה עליו תקרא לחלון הקלט עם אדם. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש, path – תמונה של האדם.

מחיקת כל הווידיג'טים הקודמים, החלפת החלון לחלון קלט של גלריה, החלפת הרקע לרקע של קלט של גלריה, כותרת- "חלון גלריה", אם אין path סימן שהמשתמש לא בחר עדיין גלריה, לכן יקראו לפונקציית הדיאלוג יחד עם גלריה. אם כן path סימן שהמשתמש בחר כבר גלריה. ישנו תא מילוי שבו המשתמש מתבקש להכניס את שם הגלריה. לאחר מכן נשלחת בקשה מהלקוח לשרת לפתוח תיקיית גלריה יחד עם תיקיית היסטוריה האחרונה שהייתה, בשם שהלקוח בחר. קוראים לפונקציה מצא את סוגי הקבצים שמחזירה מתוך path מסוים רק את הקבצים מסוג מסוים. לאחר מכן הפונקציה עוברת אחרי כל קובץ, בלולאה, ומבקשת מהשרת להכניס את כל קובץ לגלריה אצל תיקיית ההיסטוריה העכשווית ביותר שהייתה. לאחר מכן הפונקציה יוצרת מד טעינה שעולה בחלוקה שווה בין מספר התמונות, כשאר בכל פעם שהיא מקבלת הודעה מהשרת שהתמונה הוכנסה לגלריה בהצלחה המד עולה. לאחר מכן. בין העלאת התמונות, עדיין בלולאה הלקוח במקביל שומר את התמונות על המחשב של הלקוח, באופן זמני, בpath זמני ומשנה את גודלן. ברגע שכל התמונות הועלו לשרת ונשמרו אצל הלקוח, מד הטעינה נהרס. קוראים לעצם "גלריה מתגלגלת" יחד עם הpath הזמני שבו נמצאות התמונות גלריה ששמורות במחשב של המשתמש. (ראה הסבר על העצם הזה בהמשך). התמונות מוצגות יחד עם פס גלילה אנכי, ומאפסים את הרשימה שבה היו התמונות גלריה מהסוג הנכון, לפעם הבאה כי זהו משתנה גלובלי. כאשר מוצגות כל התמונות גלריה יופיע גם כפתור "הבא" שלחיצה עליו תוביל לחלון הקלט עם גלריה. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש, path – path של גלריית התמונות.

פונקציית מצא את סוגי הקבצים השונים עוברת על כל התיקיות בpath ומתוכן מוציאה תתי תיקיות, בצורה רקורסיבית, עד שמגיעה לכל הקבצים מהpath המקורי. כל פעם שמגיעה לקובץ, בודקת האם הוא קובץ מהסוג הרצוי. אם כן, מכניסה אותו לרשימה, אם לא מתעלמת. הפונקציה מחזירה את הרשימה עם הקבצים מהסוג הנכון. **משתנים:** path – path בוא יש למצוא את סוגי הקבצים הנכונים, types – הסוגים אשר צריך למצוא.

פונקציית הדיאלוג בין המשתמש לסייר הקבצים בודקת האם קיים דיאלוג כי המשתמש בחלון האדם או בחלון הגלריה או בחלון ההיסטוריה. אם בחלון האדם, היא מאפשרת למשתמש לבחור רק סוגי קבצים לפי משתנה גלובלי של סוגי קבצים קבועים, ולאחר בחירת path תקין קוראת חזרה לפונקציית האדם עם יחד עם path. אם גלריה, הפונקציה מאפשרת למשתמש לבחור רק תיקייה, וברגע שבחר directory תקין קוראת חזרה לפונקציית הגלריה יחד עם path. אם היסטוריה, הפונקציה מאפשרת למשתמש לבחור רק קבצים מסוג מותר, לפי משתנה גלובלי, והdirectory הראשוני שמוצג לאדם הוא directory של ההיסטוריה האישית במחשב של הלקוח. **משתנים:** person – בוליאני, האם מדובר בחלון קלט של אדם או לא, history – בוליאני, האם מדובר בחלון היסטוריה או לא, gallery – בוליאני, האם מדובר בחלון גלריה או לא.

מחיקת כל הווידג'טים הקודמים, יצירת רקע של תהליך הפנים, כותרת- "תמונות נכונות", אם אין עוד רשימה של התמונות הנכונות, סימן שהלקוח עדיין לא התחיל את התהליך, ולכן ישלח הלקוח בקשה לשרת שישלח לו את כל התמונות הנכונות – בתיקיה הכי עכשווית, להשוות את התמונה של האדם לכל תמונות הגלריה, ולמצוא באילו תמונות האדם נמצא, ולהחזיר ללקוח. השרת ישלח חזרה ללקוח תמונות בהם נמצא האדם. הלקוח יקרא לפונקציית הכנסת תמונות נכונות, עם שם התמונה, התמונה, ושם המשתמש. בזמן הזה יקראו לפונקציית ההמתנה. אם יש תמונות נכונות, סימן שהתהליך כבר בוצע ע"י השרת והתמונות כבר בידי הלקוח. לכן השרת יעבור על כל תמונה ישמור אותה בpath זמני, ישנה את גודלה. יהיה למשתמש מד טעינה עד שהתמונות יעלו. הלקוח יקרא לעצם "גלריה מתגלגלת" יחד עם path הזמני שבו נמצאות התמונות גלריה ששמורות במחשב של המשתמש. (ראה הסבר על העצם הזה בהמשך). התמונות מוצגות יחד עם פס גלילה אנכי, ומאפסים את הרשימה שבה היו התמונות גלריה מהסוג הנכון, לפעם הבאה כי זהו משתנה גלובלי. בסיום התהליך יהיה ללקוח שתי אפשרויות: כפתור עשה את התהליך שוב, שיקרא שוב לחלון הקלט (התהליך הנוכחי הושלם ולכן כן ישמר ע"י השרת) וכפתור המשך להיסטוריה שיקרא לפונקציית היסטוריה. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש, paths – correct images של תמונות הנכונות בהם נמצא האדם המבוקש.

מחיקת כל הווידג'טים הקודמים, החלפת הרקע לרקע של "המתן בבקשה", כותרת "חלון המתנה", בסיום התהליך תקרא, תקרא לפונקציה שקיבלת כקלט. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש,

פונקציית הכנס תמונות נכונות מכניסה לרשימה גלובלית של תמונות נכונות כל תמונה שהשרת טוען שבה נמצא האדם, זאת באמצעות יצירת התמונה בpath זמני, שמירתה (יחד עם התאריך והשעה המדויקים, למנוע כפל שמות), ולאחר מכן שולח הודעה שהתמונה הוכנסה בהצלחה, על מנת לחכות לתמונה הבאה. שם המשתמש לזיהוי בפני השרת בהודעות. **משתנים:** name – שם התמונה, data – מידע בינארי של התמונה.

מחיקת כל הווידג'טים הקודמים, שינוי הרקע לרקע היסטוריה, כותרת- "היסטוריה", אם לא סיים סימן שעוד לא הועלתה ההיסטוריה, אם גם path ההיסטוריה לא קיים, זוהי ההיסטוריה הראשונה שמועלת למשתמש לכן תיצור path קבוע של היסטוריית המשתמש, אם path קיים תשאיר את המצב כמו שהוא. אין לך תיקיות היסטוריה (עדיין לא סיים), תשלח בקשה לשרת שיעלה את השמות של תיקיות ההיסטוריה. לאחר מכן תקרא לפונקציה הזאת אם התיקיות. אם כן תיקיות היסטוריה, תפתח בpath הקבוע תיקיות לפי השמות ששלח השרת. לסיום תשלח בקשה לשרת שיתחיל לעלות קבצי היסטוריה. בכל פעם שמגיע קובץ היסטוריה הוא מועבר לפונקציה הכנס תיקיות היסטוריה, ששומרת את הקבצים. בסיום העלאת כל הקבצים השרת יודיע שאין יותר היסטוריה חדשה, ואז הלקוח יקרא לפונקציה הזו שכן סיים. אם סיים תקרא לפונקציית הדיאלוג עם היסטוריה. אם המשתמש בוחר איזשהו קובץ, הלקוח קורא לפונקציה הזו שכן סיים ועם path של התמונה. הפונקציה תשמור את התמונה בpath זמני, תשנה את גודלה, את סוגה לpng (tkinter מציג תמונות מסוג png), ותציג את התמונה. בסיום החלון יהיה למשתמש האפשרות לראות שוב את ההיסטוריה שלו, שיקרא לפונקציה הזו עם סיים, לעשות את התהליך שוב, שתקרא לפונקציית הקלט או לצאת מהאפליקציה מה שיוביל ליציאה מסודרת. event של יציאה של המשתמש באופן לא מסודר. **משתנים:** path – background של רקע המסך בהתאם למצב שבו נמצא המשתמש, history folders – השמות של תיקיות ההיסטוריה, image path – תמונה אשר המשתמש רוצה לראות, done – בוליאני, האם המשתמש סיים את העלאת ההיסטוריה או לא.

פונקציית יצירת מפתח סימטרי, שיוצרת על פי אלגוריתם RSA מפתח סימטרי פרטי, יחד עם השרת. **משתנים:** min value – ערך מינימלי, max value – ערך מקסימלי.

הוספת אפסים- מוסיפה אפסים לגודל ההודעה, כך שהשרת ידע מה גודל ההודעה, בנוסף מודיעה ללקוח אם ההודעה גדולה מידי. **משתנים:** str1 – איזושהי מחרוזת.

טיפול הודעות דואגת שהודעות גדולות מדי (מידע של קובץ תמונה) לא יודפסו אלא יודפס במקום מידע של קובץ תמונה. **משתנים:** message – ההודעה.

פונקציית בדיקה בודקת את הפאקטות שקיבל הלקוח ופועלת בהתאם. תחילה מדפיסה את טיפול ההודעות של ההודעה. לאחר מכן היא בודקת אם במקום הראשון בפאקטה יש אישור או טעות. אם אישור: יציאה- ניתוק מהתוכנית, מפתח ציבורי נשלח- שמירת המפתח כגלובל, משתמש חדש נוצר- תקרא לחלון הפתיחה עם רקע עם הודעה שהחלון נוצר בהצלחה, כניסה למערכת- תקרא לחלון ההוראות, אדם בתמונה הזו – תקרא לפונקציה הכנס תמונות נכונות, אם אין יותר תמונות נכונות- תקרא לתהליך פנים עם התמונות הנכונות שנשלחו, אם הנה תיקיות ההיסטוריה- תקרא להיסטוריה עם התיקיות הנ"ל, אם תכניס את הקובץ להיסטוריה- תקרא להכנסת קבצים להיסטוריה, אם אין עוד קבצי היסטוריה – תקרא למסך ההיסטוריה עם הקבצים שנשלחו עד כה. אם טעות: אין שם משתמש- תקרא למסך משתמש חדש עם רקע שמודיע שלא הוכנס שם משתמש, אם אין סיסמא- תקרא לפונקציה משתמש חדש עם רקע שמודיע שלא הוכנסה סיסמא, אם שם משתמש קצר מידי- תקרא לפונקציה משתמש חדש עם רקע שמודיע ששם המשתמש קצר מידי, אם סיסמא קצרה מידי- תקרא למשתמש חדש עם רקע שמודיע שהסיסמא קצרה מידי, אם שם משתמש קיים- תקרא למשתמש חדש יחד עם רגע שמודיע ששם המשתמש הזה כבר תפוס, אם שם המשתמש לא תואם לסיסמא- תקרא לכניסה לחשבון עם רקע שמודיע שהשם המשתמש או הסיסמא שגויים, אם המשתמש כרגע בשימוש- תקרא לכניסה לחשבון יחד עם רקע שאומר שאי אפשר להיכנס לאותו חשבון במקביל. **משתנים:** data – ההודעה, username – שם המשתמש, password – הסיסמא של המשתמש.

פונקציית שליחה וקבלה. אם מפתח סימטרי פרטי תצפין בעזרת המפתח הסימטרי הפרטי את ההודעות שאתה שולח לשרת. אם לא מפתח סימטרי פרטי, תצפין בעזרת RSA את ההודעה שאתה שולח כדי להחזיק יחד עם השרת מפתח ציבורי פרטי. תקרא להוספת אפסים לגודל הפאקטה שברצונך לשלוח, ומצרפת את הפאקטה המוצפנת עם ההודעה, שעוברת הפיכה למחרוזת (pickle). זאת הפאקטה החדשה. תשלח את הפאקטה: גודל והועדה מוצפנת לשרת. אם שלחת פאקטה שגודלה גדול מ3 ז"א שאתה רוצה לקבל תשובה חזרה, ובעצם לחכות להיתקע בקבלת הודעה. כאשר אתה מקבל את ההודעה תבדוק מה הגודל שלה, תחזיר אותה לרשימה (pickle), תפענח אותה בעזרת המפתח ותקרא לפונקציית הבדיקה יחד עם ההודעה. **משתנים:** data – ההודעה, use symmetrical key – בוליאני, האם להשתמש במפתח הסימטרי הפרטי או לא (אם הוא קיים או לא)

פונקציית החבא סיסמא משנה את כל התווים בתא המילוי של הסיסמא לכוכביות, ובמידה אם הם כבר כוכביות משנה אותם בחזרה. **משתנים:** תא מילוי- תא המילוי של הסיסמא עליו יבוצעו הפעולות. כפתור החבא אשר יסתיר את הסיסמא.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Communication_Settings.py	General Client	קובץ כפול בצד השרת והלקוח שנועד להבטיח שהלקוח יוכל לתקשר עם השרת.	לדאוג באופן כללי, שהלקוח פועל באופן תקין, נוחות המשתמש ותקשורת עם רציפה ותקינה עם השרת.	Client\ Communication_Settings.py

ההגדרות הכפולות של התקשרות. **משתנים:**

```
Host = '127.0.0.1' #
Port = 9998 # the port. important that the clients
and the server be in the same port
Max_Data_Size = 9 # the biggest size of data that
can be sent. 999,999,999 bits.
```

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Handle_Images.py	General Client	פונקציה מרכזית נפרדת מקובץ הפונקציות המרכזיות של השרת, שדואגת להציג הרבה תמונות, עם מד טעינה ופס גלילה.	לדאוג באופן כללי, שהלקוח פועל באופן תקין, נוחות המשתמש ותקשורת עם רציפה ותקינה עם השרת.	Client\ Handle_Images.py

שני פונקציות שדואגות לטיפול בתמונה בGUI:

עצם להצגת גלריה, מקבלת path שנמצא בו תמונות גלריה, מציגה אותם זו אחר זו, ומוסיפה פס גלילה אנכי כדי לעיין בתמונות. רקע- כרקע האפליקציה. **משתנים:** parent – החלון ה"אב", תמונות.

עצם להצגת תמונה יחידה בגדול, רקע- כרקע האפליקציה. **משתנים:** parent – החלון ה"אב", תמונות.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Encryption_And_Decryption.py	Security Client	קובץ כפול הדואג להצפין ולפענח הודעות לפי מפתח ואלגוריתם	לדאוג שמידע אישי ורגיש של הלקוח לא ייחשף	Client\ Encryption_And_Decryption.py

		בעת תקשורת עם השרת.	זהה בני הצדדים, וכך להבטיח אבטחה והבנה של ההודעות בתקשורת.	
--	--	------------------------------	---	--

כאשר לקוח מעוניין להתחבר הוא מקבל את המפתח הציבורי. לאחר מכן רק לקוח שיועד את הפרוטוקול של השרת (אם כי עדיין בעל סיכוי לניחוש\פריצה) ידע שהוא צריך ליצור בעצמו עוד מספר. יצירת המספר של הלקוח היא יצירת מספר רנדומלי (באמצעות Random) בין המספר הציבורי השני שקיבל לבין מספר היכולת הרצה של המחשב שלך (ככל שהכוח המעבד שלך קטן יותר כך המספר יקטן). מכיוון שאני עובד במחשב ביתי, ואין ברשותי כוח מעבד כמו חברות אבטחה גדולות שמגיעות גם למספרים של 50 ו-60 ספרות, מספר ההגבלה שלי הוא 150000. הגעתי למספר זה באמצעות לולאה ששינתי כל פעם את המספר עד שהגעתי למספר ההגבלה הגדול ביותר וזמן ההרצה הקטן ביותר (לא יותר מכמה שניות במקרה הגרוע ביותר). לאחר יצירת המספר הנ"ל הלקוח מעלה את המספר שיצר בחזקת המספר הראשון הציבורי והשאירית של התוצאה מהמספר השני הציבורי תיתן את המפתח הסימטרי הפרטי. הלקוח שולח את המספר הרנדומלי (לא הפרטי) שמצא לשרת.

השרת מקבל את המספר מעלה אותו בחזקת המספר הפרטי של השרת. לתוצאה הוא מחפש שארית מהמספר השני הציבורי. והתוצאה היא המפתח הסימטרי הפרטי. השרת שומר ברשימה את הלקוח ואת המפתח הסימטרי פרטי שלו, כמובן לכל לקוח מפתח סימטרי פרטי אחר. כך יודע השרת כשאר הוא מדבר עם לקוח איך להצפין ולפענח הודעות שנשלחות ביניהן כך שרק הם ידעו.

כך, בלי לשלוח ברשת את המפתח שבו יצפינו הלקוח והשרת באותה שיטת הצפנה (שאפרט בהמשך) ללקוח ולשרת יש "ביד" את אותו מספר. כך מובטח שגם אם גורם שלישי יאזין ברשת, בלי המספר הפרטי של השרת d הוא לא יוכל לדעת מהו המפתח הציבורי הפרטי של הלקוח והשרת.

הצפנת משנה: לאחר שלשרת יש מפתח סימטרי פרטי אם כל לקוח יש להסכים שצד הלקוח והשרת לגבי אלגוריתם הצפנה ופיענוח שרק מחזיק מפתח מסוים ידע מה המידע בהודעה. אלגוריתם ההצפנה בנוי כך: פונקציה המקבלת מפתח מסוים, ופאקטה (הודעה) מסוג רשימה. הפונקציה עוברת על כל הודעה ברשימה בנפרד ומצפינה אותו בנפרד.

הודעה מוצפנת כך: עוברים על כל האותיות של ההודעה. לכל אות מעבירים למספר Unicode שלו (באמצעות פונקציית ord). את המספר הנ"ל הופכים (נגיד 1546 הופך ל6451). מחסירים ממנו 100 עד שנמצא בין 0 ל-255, בינתיים ישנו מונה (ערכו הראשוני 40) שעולה, בכל החסרה של המספר ההפוך, במפתח ההצפנה. מכניסים לתוך רשימה את המספר ההפוך ואת המונה. לאחר מכן עוברים לאות הבאה. בסופו של דבר ישנה רשימה המייצגת מילה עם מספר הפוך ומונה וכך הלאה. לאחר מכן, כל איבר ברשימה משונה למספר ascii שלו. כך בעצם לא ניתן לדעת ברוב מוחלט של המקרים איזה מספר ייצג התוצאה כי המספר המקורי היה גדול שלא ניתן כמעט לשחזר אותו אחורנית להיות מספר. לאחר מכן, מחברים לstring אחד את כל הרשימה וכל הרשימה הנ"ל מייצגת מילה אחת בפאקטה. כך עוברים על כל המילים בפאקטה ומצפינים את כולם. רק למי שיש את המפתח יהיה מסוגל להעביר את האותיות הascii למקור, שכן הוא יודע כמה (בעזרת צירוף המונה ליד כל אות) פעמים להכפיל את המפתח ואיתו להחזיר לאחורנית לאות. פונקציית הפיענוח פשוט הפוכה מפונקציית ההצפנה וכך בעצם מפענחים ומצפינים חזרה למקור את כל ההודעות בלי למחוק מידע כלל. בעצם גורם המסתכל בפאקטות העוברות ברשת יראה רק

מידע מוצפן. גם בעזרת brute force לא ימצא מהי הפאקטה המקורית כי אין ברשותנו את המפתח. כאמור הוא יכול לנחש פעם את המפתח ועליו לנסות brute force, אבל כמובן דבר זה ייקח לו שנים. חיסרון יחיד קטן, שבו נתקלתי והצלחתי לעקוף זה שאתה חייב לשלוח תווים שיש להם Unicode. זה הפריע לי כאשר ניסיתי לשלוח מידע של תמונה שבנוי מאלפי תווים אבל זוהי בעיה ידועה באינטרנט אליה מצאתי פתרון. בנוסף לכל הודעה, בגלל שהיא רשימה, עוברת pickle, שהופך אותה לstring ארוך. דבר שמקשה עוד יותר על גורם שלישי עוין.

וכך, מועברות באופן מאובטח לחלוטין תחילה באמצעות RSA מפתחות סימטריים פרטיים ואז באמצעות אלגוריתם ההצפנה הנ"ל מידע ופאקטות רגישים וסודיים. בעצם רק השרת והלקוח יודעים את ההודעות שהם שולחים זה לזה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\blank.jpg	עיצוב גרפי של האפליקציה.	רקע ריק, למקרה הצורך.	Backgrounds	blank.jpg

רקע ריק, למקרה הצורך.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\detection_face.jpg	עיצוב גרפי של האפליקציה.	רקע בו יוצג האדם אותו רוצים לחפש.	Backgrounds	detection_face.jpg

רקע בו יוצג האדם אותו רוצים לחפש.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\exists_user.jpg	עיצוב גרפי של האפליקציה.	רקע להכנסת הפרטים לכניסה לחשבון קיים.	Backgrounds	exists_user.jpg

רקע להכנסת הפרטים לכניסה לחשבון קיים.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\gallery.jpg	עיצוב גרפי של האפליקציה.	רקע להצגת גלריית התמונות מתוכו חיפש המשתמש אדם מסוים.	Backgrounds	gallery.jpg

רקע להצגת גלריית התמונות מתוכו חיפש המשתמש אדם מסוים.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Client\Design\Backgrounds\history.jpg	עיצוב גרפי של האפליקציה.	רקע להצגת ההיסטוריה של המשתמש.	Backgrounds	history.jpg

רקע להצגת ההיסטוריה של המשתמש.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
How_to_use.jpg	Backgrounds	רקע וחלון המסביר כיצד להשתמש ומה השימוש של האפליקציה.	עיצוב גרפי של האפליקציה.	Client\Design\Backgrounds\How_to_use.jpg

רקע וחלון המסביר כיצד להשתמש ומה השימוש של האפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
inputs.jpg	Backgrounds	רקע לחלון קלט הנתונים של המשתמש.	עיצוב גרפי של האפליקציה.	Client\Design\Backgrounds\inputs.jpg

רקע לחלון קלט הנתונים של המשתמש.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Main.jpg	Backgrounds	רקע הפתיחה של האפליקציה.	עיצוב גרפי של האפליקציה.	Client\Design\Backgrounds\Main.jpg

רקע הפתיחה של האפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
new_user.jpg	Backgrounds	רקע להרשמת משתמש חדש לאפליקציה.	עיצוב גרפי של האפליקציה.	Client\Design\Backgrounds\new_user.jpg

רקע להרשמת משתמש חדש לאפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
register_user_main.jpg	Backgrounds	רקע הפתיחה עם הודע שההתחברות של המשתמש לאפליקציה בוצעה בהצלחה.	עיצוב גרפי של האפליקציה.	Client\Design\Backgrounds\register_user_main.jpg

רקע הפתיחה עם הודע שההתחברות של המשתמש לאפליקציה בוצעה בהצלחה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
----------	-------	-------------	--------------	-------------

wait.jpg	Backgrounds	רקע חלון המודיע למשתמש להמתין.	עיצוב גרפי של האפליקציה.	Client\Design\Backgrounds\wait.jpg
----------	-------------	--------------------------------	--------------------------	------------------------------------

רקע והקפצת הודעת שגיאה שהמשתמש הכניס נתונים שגויים בעת ניסיון התחברות למשתמש. קיים באפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
no_face.jpg	Error Detection Faces	רקע והקפצת הודעת שגיאה שהמשתמש הכניס תמונה ללא פנים.	עיצוב גרפי של האפליקציה והדרכת המשתמש אחרי התחלת האפליקציה.	Client\Design\Error Detection Faces\ no_face.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס תמונה ללא פנים.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
to_many_faces.jpg	Error Detection Faces	רקע והקפצת הודעת שגיאה שהמשתמש הכניס תמונה עם יותר מפרצוף אחד.	עיצוב גרפי של האפליקציה והדרכת המשתמש אחרי התחלת האפליקציה.	Client\Design\Error Detection Face\ to_many_faces.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס תמונה עם יותר מפרצוף אחד.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
no_gallery.jpg	Error Detection Faces	רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס גלריה כלל.	עיצוב גרפי של האפליקציה והדרכת המשתמש אחרי התחלת האפליקציה.	Client\Design\Error Detection Face\ no_gallery.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס גלריה כלל.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
wrong_username_or_password.jpg	Start Errors	רקע והקפצת הודעת שגיאה שהמשתמש הכניס נתונים שגויים בעת ניסיון התחברות למשתמש. קיים באפליקציה.	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	Client\Design\wrong_username_or_password.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס נתונים שגויים בעת ניסיון התחברות למשתמש. קיים באפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
no_password.jpg	Start Errors	רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס סיסמא בעת הרשמה לאפליקציה.	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	Client\Design\no_password.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס סיסמא בעת הרשמה לאפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
no_username.jpg	Start Errors	רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש בעת הרשמה לאפליקציה.	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	Client\Design\no_username.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש בעת הרשמה לאפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
password_short.jpg	Start Errors	רקע והקפצת הודעת שגיאה שהמשתמש הכניס סיסמא קצרה מדי בעת הרשמה לאפליקציה.	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	Client\Design\password_short.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס סיסמא קצרה מדי בעת הרשמה לאפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
username_already_in_use.jpg	Start Errors	רקע והקפצת הודעת שגיאה שהמשתמש הכניס שם משתמש שכבר תפוס בעת הרשמה לאפליקציה.	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	Client\Design\ username_already_in_use.jpg

רקע והקפצת הודעת שגיאה שהמשתמש הכניס שם משתמש שכבר תפוס בעת הרשמה לאפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
username_short.jpg	Start Errors	רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש קצר מידי בעת הרשמה לאפליקציה.	עיצוב גרפי של האפליקציה והדרכת המשתמש בתחילת האפליקציה.	Client\Design\ username_short.jpg

רקע והקפצת הודעת שגיאה שהמשתמש לא הכניס שם משתמש קצר מידי בעת הרשמה לאפליקציה.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Project_File.docx	Documentation	להסביר הרחבה על כל מה שקשור לפרויקט הנ"ל.	תיעוד נרחב של רעיון, מימוש, רקע והסבר של האפליקציה.	Documentation\ Project_File.docx

להסביר הרחבה על כל מה שקשור לפרויקט הנ"ל.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Idea_Suggestion_Doc.docx	Documentation	הצעת הרעיון של הפרויקט הנ"ל.	תיעוד נרחב של רעיון, מימוש, רקע והסבר של האפליקציה.	Documentation\ Specification_Doc.docx

הצעת הרעיון של הפרויקט הנ"ל.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
----------	-------	-------------	--------------	-------------

Specification_Doc.docx	Documentation	אלגוריתמיקה ראשונית של הפרויקט הנ"ל.	תיעוד נרחב של רעיון, מימוש, רקע והסבר של האפליקציה.	Documentation\ Specification_Doc.docx
------------------------	---------------	--------------------------------------	---	---------------------------------------

אלגוריתמיקה ראשונית של הפרויקט הנ"ל.

שם הקובץ	מחלקה	תפקיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Communication_Settings.py	General Server	קובץ כפול בצד השרת והלקוח שנועד להבטיח שהלקוח יוכל לתקשר עם הלקוח.	לדאוג לתפקוד תקין של השרת, לענות בהתאם ללקוחותיו.	Server\ Communication_Settings.py

ההגדרות הכפולות של התקשרות. **משתנים:**

```
Host = '127.0.0.1' #
Port = 9998 # the port. important that the clients
and the server be in the same port
Max_Data_Size = 9 # the biggest size of data that
can be sent. 999,999,999 bits.
```

שם הקובץ	מחלקה	תפקיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Run_Server.py	General Server	הרצה תמידית של השרת, נתינת נתונים נכונים ותקניי לשרת, דואג לתמידות נכונה ותקינה של השרת.	לדאוג לתפקוד תקין של השרת, לענות בהתאם ללקוחותיו.	Server\ Run_Server.py

יוצר קובץ DB אם לא קיים, אם קיים רק פותח אותו. מאפס את האקטיביות של כל המשתמשים, ליתר ביטחון, יוצר שרת חדש בלולאה מתמדת.

שם הקובץ	מחלקה	תפקיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Server_Functions.py	General Server	פונקציות סנכרון החזרת/קבלת מידע מלקוחות, יחד עם פונקציות אלגוריתמיות לשמירת תמונות והיסטוריה.	לדאוג לתפקוד תקין של השרת, לענות בהתאם ללקוחותיו.	Server\ Server_Functions.py

פונקציית יצירת שרת, יוצרת שרת עם כתובת מסויימת שמוסכת על הלקוחות, יכולה לקלוט עד 5 לקוחות בו זמנית (listen), וקוראת לייצור המפתח הפרטי של השרת באמצעות RSA. בסיום כל התהליך, קם שרת מרובה לקוחות.

פונקציית השרת האקטיבי שמורצת בלולאה אינסופית קולטת לקוחות חדשים, מוציאים לקוחות בעייתיים, ומפרידה בין לקוחות שמחכים לקבלת הודעה משרת. השרת נכתב באמצעות select. אם הלקוח הוא עדיין socket, סימן שעוד לא בוצע חיבור ולכן יש לחבר בין השרת ללקוח. בחיבור השני הוא כבר לא socket ולקוח לכל דבר. בכל מקרה השרת מחכה להודעה מהלקוח. כל הודעה עוברת לפונקציה שליחת הודעה בהתאם לבקשת הלקוח.

פונקציית שליחת הודעות בהתאם לבקשת הלקוח מחכה להודעה מהלקוח. ברגע שמגיעה הודעה היא פועלת בהתאם. אם זוהי הודעת בקשה: 'ציאה' – ניתוק הלקוח באופן נקי: להפוך אותו ללא אקטיבי ולשמור את המידע שלו וכו' (ראה ארכיטקטורת רשת, עמוד 11).

פונקציית יצירת מפתח סימטרי, שיוצרת על פי אלגוריתם RSA מפתח סימטרי פרטי, יחד עם השרת. **משתנים:** min value – ערך מינימלי, max value – ערך מקסימלי.

הוספת אפסים- מוסיפה אפסים לגודל ההודעה, כך שהשרת ידע מה גודל ההודעה, בנוסף מודיעה ללקוח אם ההודעה גדולה מידי. **משתנים:** str1 – איזושהי מחרוזת.

טיפול הודעות דואגת שהודעות גדולות מדי (מידע של קובץ תמונה) לא יודפסו אלא יודפס במקום מידע של קובץ תמונה. **משתנים:** message – ההודעה.

פונקציית למצוא רדאר שמדפיסה את נתוני הלקוח ממנו התקבלה ההודעה. **משתנים:** לקוח.

פונקציית יצירת גלריה יוצרת בpath של המשתמש הנוכחי של האדם הנוכחי path של גלריית תמונות שתהיה מוכנה להעלות תמונות. שולחת ללקוח שהגלריה נוצרה בהצלחה. **משתנים:** שם משתמש (כדי להוציא את הpath שמירת נתונים של המשתמש), שם הגלריה, מבנה הנתונים.

מחיקת מידע לא נחוץ, עוברת על כל המידע שברשות משתמש מסוים ובודקת האם שלושת התיקיות של אדם, גלריה ותמונות נכונות קיים. אם לא השרת מוחק תיקייה זו. **משתנים:** שם המשתמש, מבנה הנתונים.

יצירת תיקייה מקבלת שם של תיקייה ויוצרת תיקיית תהליך חדשה בpath המידע של המשתמש. בנוסף גם מקבלת תמונה של אדם אותו מחפשים. בודקת ששם התיקייה לא נמצא שוב ואם כן מוסיפה את הספרה 1 לסיום, ואחר כך 2 וכך הלאה. שולחת הודעה לשרת שהתיקייה נוצרה בהצלחה. **משתנים:** שם משתמש, התמונה של האדם, שם של התיקייה, מבנה הנתונים.

בדיקת משתמש חדש בודקת האם שם המשתמש והסיסמא תקינים: מעל 4 תווים כל אחד, שם המשתמש לא קיים כבר, הוקלד איזושהי שם משתמש ואיזושהי סיסמא. במידה וישנה טעות השרת מודיעה על הטעות ללקוח. אם הכל תקני השרת מודיע ללקוח שהחשבון נוצר בהצלחה. **משתנים:** שם משתמש אתו רוצים לפתוח את החשבון, סיסמא של שם המשתמש, מבנה נתונים.

בדיקת משתמש קיים בודקת האם שם המשתמש תואם את הסיסמא. אם לא שולח הודעה ללקוח ששם המשתמש או הסיסמא אינם נכונים. בנוסף בודק שהמשתמש לא אקטיבי, ושאינן ניסיון חיבור לאותו משתמש במקביל. אם יש כזה ניסיון שולח הודעה שלא ניתן להתחבר לאותו משתמש במקביל. אם הכל תקין שולח הודעה שהכניסה לחשבון בוצעה בהצלחה.

משתנים: שם משתמש אתו רוצים להיכנס את החשבון, סיסמא של שם המשתמש, מבנה נתונים.

הכנסת אדם לגלריה מקבל אדם, ומכניסה אותו לגלריה שנמצאת בתיקייה הכי עכשווית. מחזירה הודעה שהאדם הוכנס בהצלחה. **משתנים:** שם משתמש, התמונה, השם של התמונה, מבנה נתונים.

מציאת האדם שולחת למשתמש את התמונות בהם האדם בתיקייה הכי עכשווית המבוקש נמצא בגלריה של אותה תיקייה. הוא נותן את paths של הגלריה והאדם לפונקציית מציאת אדם שמחזיר לו רשימה של התמונות בהם נמצא האדם המבוקש. מתוך הרשימה הזו השרת שולח תמונה כל פעם ללקוח, ומחכה מעט זמן כדי שהלקוח יעשה תהליך בצד שלו. כאשר הרשימה נגמרת שולח השרת שנגמרו כל התמונות בהם האדם נמצא לשרת. **משתנים:** הלקוח, שם המשתמש, מבנה הנתונים.

העברת שמות תיקיות. השרת יקבל את path של הלקוח גם בעזרת פרטיו וטבלת sql שברשותו. כך ידע לאן לגשת ולהוציא את כל תיקיות ההיסטוריה שקיימות לאותו משתמש במחשב לו. תיקיות אלו מעולות ישר לשרת, אך לא ישר ללקוח. בעזרת חיסור הרשימות ידע השרת אילו תיקיות עוד לא העלה ללקוח ואת שמם ישלח ללקוח עם הוראה ליצור תיקיות אלו בpath ההיסטוריה של הלקוח. **משתנים:** לקוח, שם משתמש, מבנה נתונים.

שליחת קבצי היסטוריה שולחת ללקוח את כל קבצי ההיסטוריה בתיקיות ההיסטוריה החדשות, כדי שישמור אותם אצלם. הוא שולח הודעה בה שם התיקייה, התת-תיקייה, שם הקובץ, מידע בינארי של הקובץ. בסיום שולח השרת ללקוח שאין יותר קבצים. **משתנים:** הלקוח, שם המשתמש, התיקיות.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Database.py	העברת נתונים למבנה הנתונים בהתאם לבקשת השרת, באופן תקין, נכון ומסודר.	הכנסה, הוצאה וכל טיפול שהוא במבנה הנתונים הקיים, לצורך תפקוד תקין של השרת.	Database	Database.py

פעולות בנוגע לטבלת sql כגון להפוך משתמש לאקטיבי, ללא אקטיבי, לתת למשתמש path היסטוריה במחשב של השרת ולשמור אחת כזו, יצירת DB חדש\לפתוח את הקובץ הקיים ולפעול בו, לקחת מידע ברשימות מן עמודות, להניס שם משתמש עם סיסמא, למחוק משתמש, להשיג את path ההיסטוריה של המשתמש, להכניס את תיקיות ההיסטוריה שכבר הועלו אל הלקוח של המשתמש, להכניס את תיקיות ההיסטוריה שכבר הועלו אל הלקוח של המשתמש, להחזיר את תיקיות ההיסטוריה שכבר הועלו אל הלקוח של המשתמש, למחוק את הטבלה, להחזיר את כל הנתונים בטבלה, מחזירה true אם הטבלה מורצת מcmd או מpython. **משתנים:** history paths – איפה לפתוח תיקיות היסטוריה חדשות, Server path – איפה השרת מורץ database path – איפה מבנה הנתונים הקיים\ איפה לפתוח אחד חדש.

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Detect_faces.py	Detection and Compression Faces	אלגוריתמיקה של מציאת והשוואה של פרצופים בין תמונות שונות, תוך כדי שימוש במכונה לומדת מאומנת.	לאתר ולהשוות פנים מבין תמונות שונות.	Server\ Detect_faces.py

פונקציית האם פנים מקבלת תמונה אשר אותה מכניסה למכונה שמכבר מזהה מאזכרים של פנים כמו: עיניים, אף, פה, תווי פנים וכדומה. המכונה עוברת על כל חלקי התמונה, סורקת, וכך יודעת כמה פרצופים יש בתמונה. המכונה עושה זאת משום שכבר אומנה על מיליוני תמונות של פרצופים ויכולה לזהות במהירות ובדיוק רב כמה פרצופים יש בתמונה. המכונה המאומנת מחזירה שקר יחד עם הודעה "יותר מידי פרצופים" כאשר יש יותר מפרצוף אחד בתמונה, מחזירה שקר יחד עם הודעה "אין פרצופים" כאשר אין פרצופים בתמונה. המכונה מחזירה אמת ושהכל תקין אם יש פרצוף אחד בתמונה. זה לחלק אחד בתוכנית כאשר המשתמש מכניס תמונה של לקלט של אדם, ועל השרת לבדוק האם יש פרצוף אחד בתמונה או לא. **משתנים:** קובץ תמונה.

החלק השני קשור להשוואה בין תמונה לגלריית תמונות והשאלה היא האם האדם בתמונה הזו הוא אותו אדם בתמונה השנייה. לשם כך יש את פונקציית השווה פנים שמקבלת תמונה של אדם וגלריה. הפונקציה בפועל בודקת כל פעם בין שתי תמונות, בין האדם המבוקש לתמונה אחת מהגלריה עד שהיא עוברת על כל תמונות הגלריה. גם פה מדובר במכונה לומדת, שקיבלת מיליוני lists ו tuples של אנשים זהים עד שידעה לייצר את מספר הדמיון בין שתי תמונות. כמובן, ככל שהתמונה יותר ברורה, חדה ובהירה אחוזי הדמיון עולים. אני מריץ כמה תמונות במקביל על מנת לייעל את זמן העבודה, מספר התמונות שמורצות בו זמנית שווה למספר CPUs שיש לך על המחשב (לי יש 4). המכונה שומרת את התמונות בהם יש אחוזי דמיון גבוהים, וגם פה נתתי למכונה לא רף אחוז דמיון גבוה כדי שלא יתפספו תמונות נכונות. המכונה מחזירה את התמונות בהם האדם נמצא. **משתנים:** תיקיית תמונות – גלריה, תמונה של אדם, מספר CPUs (מוגדר אצלי כ-4), רף דמיון מינימלי (מוגדר אצלי כ-0.6), הראה דמיון בין אנשים (בוליאני).

שם הקובץ	מחלקה	תקפיד הקובץ	תפקיד המחלקה	מיקום הקובץ
Encryption_And_Decryption.py	Security Server	קובץ כפול הדואג להצפין ולפענח הודעות לפי מפתח ואלגוריתם זהה בני הצדדים, וכך להבטיח אבטחה והבנה של ההודעות בתקשורת.	לדאוג שמידע אישי ורגיש של הלקוח לא ייחשף בעת תקשורת עם השרת.	Server\ Encryption_And_Decryption.py

הצפנת משנה: לאחר שלשרת יש מפתח סימטרי פרטי אם כל לקוח יש להסכים שצד הלקוח והשרת לגבי אלגוריתם הצפנה ופיענוח שרק מחזיק מפתח מסוים ידע מה המידע בהודעה. אלגוריתם ההצפנה בנוי כך: פונקציה המקבלת מפתח מסוים, ופאקטה (הודעה) מסוג רשימה. הפונקציה עוברת על כל הודעה ברשימה בנפרד ומצפינה אותו בנפרד.

הודעה מוצפנת כך: עוברים על כל האותיות של ההודעה. לכל אות מעבירים למספר Unicode שלו (באמצעות פונקציית ord). את המספר הנ"ל הופכים (נגיד 1546 הופך ל6451). מחסירים ממנו 100 עד שנמצא בין 0 ל255, בינתיים ישנו מונה (ערכו הראשוני 40) שעולה, בכל החסרה של המספר ההפוך, במפתח ההצפנה. מכניסים לתוך רשימה את המספר ההפוך ואת המונה. לאחר מכן עוברים לאות הבאה. בסופו של דבר ישנה רשימה המייצגת מילה עם מספר הפוך ומונה וכך הלאה. לאחר מכן, כל איבר ברשימה משונה למספר ascii שלו. כך בעצם לא ניתן לדעת ברוב מוחלט של המקרים איזה מספר ייצג התוצאה כי המספר המקורי היה גדול שלא ניתן כמעט לשחזר אותו אחורנית להיות מספר. לאחר מכן, מחברים לstring אחד את כל הרשימה וכל הרשימה הנ"ל מייצגת מילה אחת בפאקטה. כך עוברים על כל המילים בפאקטה ומצפינים את כולם. רק למי שיש את המפתח יהיה מסוגל להעביר את האותיות הascii למקור, שכן הוא יודע כמה (בעזרת צירוף המונה ליד כל אות) פעמים להכפיל את המפתח ואיתו להחזיר לאחורנית לאות. פונקציית הפיענוח פשוט הפוכה מפונקציית ההצפנה וכך בעצם מפענחים ומצפינים חזרה למקור את כל ההודעות בלי למחוק מידע כלל. בעצם גורם המסתכל בפאקטות העוברות ברשת יראה רק מידע מוצפן. גם בעזרת brute force לא ימצא מהי הפאקטה המקורית כי אין ברשותו את המפתח. כאמור הוא יכול לנחש פעם את המפתח ועליו לנסות brute force, אבל כמובן דבר זה ייקח לו שנים. חיסרון יחיד קטן, שבו נתקלתי והצלחתי לעקוף זה שאתה חייב לשלוח תווים שיש להם Unicode. זה הפריע לי כאשר ניסיתי לשלוח מידע של תמונה שבנוי מאלפי תווים אבל זוהי בעיה ידועה באינטרנט אליה מצאתי פתרון. בנוסף לכל הודעה, בגלל שהיא רשימה, עוברת pickle, שהופך אותה לstring ארוך. דבר שמקשה עוד יותר על גורם שלישי עוין.

וכך, מועברות באופן מאובטח לחלוטין תחילה באמצעות RSA מפתחות סימטריים פרטיים ואז באמצעות אלגוריתם ההצפנה הנ"ל מידע ופאקטות רגישים וסודיים. בעצם רק השרת והלקוח יודעים את ההודעות שהם שולחים זה לזה.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\ Preparation_For_Encryption.py	לדאוג שמידע אישי ורגיש של הלקוח לא ייחשף בעת תקשורת עם השרת.	דואג ליצירת מספרים ומפתחות ציבוריים ופרטיים סימטריים עם הלקוח, דבר שדואג להצפנה ופיענוח נכון, תקשורת תקינה ומאובטחת.	Security Server	Preparation_For_Encryption.py

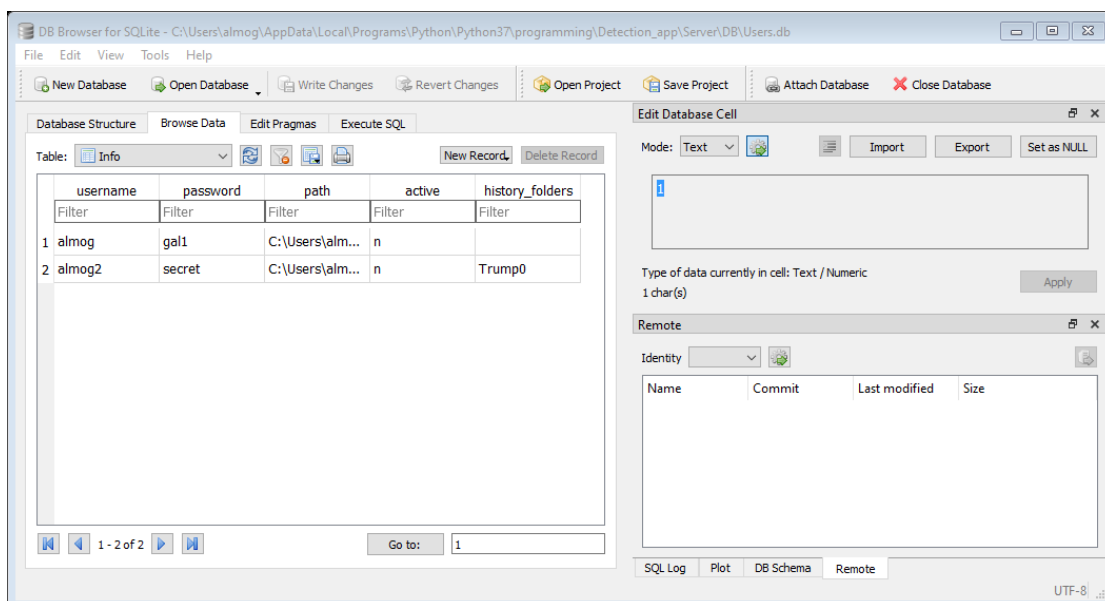
RSA: כאשר השרת נהיה אקטיבי, הוא קורא לפונקציה שיוצרת את המפתח הציבורי. המפתח הציבורי נשמר כמשתנה גלובלי לאורך כל זמן הרצת השרת.

המפתח הציבורי בנוי משני מספרים: מספר ראשון - n - מכפלת שני מספרים ראשוניים רנדומליים. m- ביצוע פונקציית אולר (Euler) על המספר הראשון. מספר שני – לאחר יצירת m התחלה של חיפוש מספר החל מ 5 (כל מספר ראשוני קטן עובד, בחרתי ב 5) של מספר שגם לא זוגי וגם זר לחלוטין (אין להם מחלק משותף) מציאת המספר הנ"ל הינו המספר השני - e. הפתח הציבורי, כשמו הוא, נשלח לכל לקוח שמעוניין בו. לאחר מכן השרת יוצר מפתח פרטי שרק הוא יודע, זאת באמצעות אלגוריתם פשוט לשרת, אך לניחוש פורץ מאוד מסובך. אני בחרתי ליצור מספר k שיהיה שווה לאחד ויגדיל את עצמו באחד בלולאה. בכל פעם בלולאה אני קורא לפונקציה בה אני מכפיל את k במ מוסיף להם אחד ומחלק ב e. אם התוצאה שלמה היא המספר הפרטי של השרת - d.

השרת מקבל את המספר מעלה אותו בחזקת המספר הפרטי של השרת. לתוצאה הוא מחפש שארית מהמספר השני הציבורי. והתוצאה היא המפתח הסימטרי הפרטי. השרת שומר ברשימה את הלקוח ואת המפתח הסימטרי פרטי שלו, כמובן לכל לקוח מפתח סימטרי פרטי אחר. כך יודע השרת כשאר הוא מדבר עם לקוח איך להצפין ולפענח הודעות שנשלחות ביניהן כך שרק הם ידעו.

כך, בלי לשלוח ברשת את המפתח שבו יצפינו הלקוח והשרת באותה שיטת הצפנה (שאפרט בהמשך) ללקוח ולשרת יש "ביד" את אותו מספר. כך מובטח שגם אם גורם שלישי יאזין ברשת, בלי המספר הפרטי של השרת d הוא לא יוכל לדעת מהו המפתח הציבורי הפרטי של הלקוח והשרת.

מיקום הקובץ	תפקיד המחלקה	תקפיד הקובץ	מחלקה	שם הקובץ
Server\DB\Users.db	שמירת נתוני הלקוחות באופן מסודר ותמידי אך ניתן לשינויים.	שמירת נתונים במסד נתונים כלשהו, באופן תמידי, נוח המאפשר שינויים.	Database	Users.db



1. שמירת נתונים במסד נתונים כלשהו, באופן תמידי, נוח המאפשר שינויים. בעמודה שם משתמש יהיה שם המשתמש של המשתמש. שם משתמש תקין וייחודי. באמצעות sql ניתן לקבוע שעמודה זו תהיה ייחודית. משתנים מסוג text (string בפיתון).
2. בסיסמא תהיינה סיסמת המשתמש. סיסמת המשתמש תהיה תקינה ותמיד תהיה קשורה לשם המשתמש תחתיו נכנסה. משתנים מסוג text.
3. במיקום יהיה מיקום שמירת תמונות המשתמש אצל השרת. כך השרת ידע לאיפה לגשת כאשר הוא צריך מידע הקשור ללקוח, למשל על מנת לעלות את היסטוריית הלקוח יש לדעת איפה היא נמצאת- במיקום שלו, וכך ניתן לגשת אליה. משתנה מסוג text.
4. במחבר יהיה כן או לא לפי האם המשתמש מחובר או לא. כאשר ישנה התנתקות מכל סוג שהיא השרת שם לב שהלקוח מכתובת מסויימת לא מחובר\לא מגיב וכך השרת יודע שהלקוח התנתק מאיזושהי סיבה. כך בעצם מעדכן השרת את העמודה. כך גם ניתן לוודא שאף לקוח לא מתחבר למשתמש שלו במקביל ולמנוע את קריסת השרת. משתנה מסוג text.
5. בתיקיות היסטוריה יהיה תיקיות ההיסטוריה שהשרת כן העלה ללקוח. כך השרת יודע אילו תיקיות חדשות נוספו מהפעם הקודמת שהוא העלה היסטוריה כי בכל העלאת היסטוריה השרת מעדכן בהתאם את העמודה הנ"ל. כך בעצם מובטח יעילות מעל הבחינות כשאר השרת יודע אילו תיקיות היסטוריה עליו לעלות. משתנה מסוג text.

בסיס הנתונים:

בפרויקט יש בסיס נתונים הנכתב באמצעות sql בפייתון. בסיס נתונים זה ישתמש בסוג sqlight3. בסיס נתונים הפנימי יהיה טבלה, בעלת עמודות ושורות, בה ישמור את הנתונים באופן מאובטח ומסודר. בטבלה חמש עמודות: שם משתמש, סיסמא, מיקום מחובר, תיקיות היסטוריה.

1. בעמודה שם משתמש יהיה שם המשתמש של המשתמש. שם משתמש תקין וייחודי. באמצעות sql ניתן לקבוע שעמודה זו תהיה ייחודית. משתנים מסוג text (string בפייתון).
 2. בסיסמא תהיינה סיסמת המשתמש. סיסמת המשתמש תהיה תקינה ותמיד תהיה קשורה לשם המשתמש תחתיו נכנסה. משתנים מסוג text.
 3. במיקום יהיה מיקום שמירת תמונות המשתמש אצל השרת. כך השרת ידע לאיפה לגשת כאשר הוא צריך מידע הקשור ללקוח, למשל על מנת לעלות את היסטוריית הלקוח יש לדעת איפה היא נמצאת- במיקום שלו, וכך ניתן לגשת אליה. משתנה מסוג text.
 4. במחובר יהיה כן או לא לפי האם המשתמש מחובר או לא. כאשר ישנה התנתקות מכל סוג שהיא השרת שם לב שהלקוח מכתובת מסויימת לא מחובר\לא מגיב וכך השרת יודע שהלקוח התנתק מאיזושהי סיבה. כך בעצם מעדכן השרת את העמודה. כך גם ניתן לוודא שאף לקוח לא מתחבר למשתמש שלו במקביל ולמנוע את קריסת השרת. משתנה מסוג text.
 5. בתיקיות היסטוריה יהיה תיקיות ההיסטוריה שהשרת כן העלה ללקוח. כך השרת יודע אילו תיקיות חדשות נוספו מהפעם הקודמת שהוא העלה היסטוריה כי בכל העלאת היסטוריה השרת מעדכן בהתאם את העמודה הנ"ל. כך בעצם מובטח יעילות מעל הבחינות כשאר השרת יודע אילו תיקיות היסטוריה עליו לעלות. משתנה מסוג text.
- כל שורה תהייה פיסת מידע - text. הסיבה מאוחרי לשמור את המידע בפיסות מידע הוא כי אין מערכים בsqlight3 ולכן אי אפשר לעשות מערך של התמונות של המשתמש (רק לעלות תמונה אחת כל פעם – מאוד לא יעיל). ובשביל לפתור בעיה זו בדרך היעלה ביותר נאלצתי לפתוח במחשב השרת תיקייה לכל שם משתמש, התיקרא על שמו הייחודית ושם לשמור את התמונות. בשביל לא להגיע למצב שתמונה תישמר עם אותו השם פעמיים אני אוסיף את התאריך והשעה המדויקת של שמירת התמונה על שרת המחשב.

תיקיות היסטוריה	מחובר	מיקום	סיסמא	שם משתמש
Yossi, Miriam, Dana	n	Db\Almog	1234	Almog
	y	Db\Dani1	Hello123	Dani1

לדוגמא אלמוג נכנס ל"היסטוריה אישית". התוכנה תחפש במיקום Db\Almog את כל התיקיות של האנשים שאלמוג רצה לחפש. תחת כל תיקייה יהיו האנשים שנמצאו להיות מתאימים לשם התיקייה הגלריה של האנשים והתמונה המקורית של אותו אדם. דוגמא נוספת היא כאשר השרת יפול כל המשתמשים יהפכו למנותקים וכך לא ינעלו משתמשים. דוגמא נוספת היא שלא תהיינה תיקייה עם אותו שם בשל העובדה שהתיקיות הם לפי שמות המשתמש שהם דבר ייחודי לפי db, וגם התמונות יישמרו תחת שם התמונה ותאריך ושעה של תמונה.

אלמוג גל. Face Founder.

ביבליוגרפיה

Websites:

Google

Stack Overflow

Wikipedia

סיכום אישי \ רפלקציה

עבורי העבודה על הפרויקט הייתה מעשירה ומעצימה מבחינת ידע תכנותי מודרני (למשל כתיבת קוד טובה בפייתון). התחלתי את הפרויקט עם לא הרבה ידע לגבי שרת-לקוח, רשת, שפת פייתון GUI. אם כי, בסוף התהליך אני יודע טוב את כולם. אז במבחן התוצאה הפרויקט העשיר רבות את הידע שלי במחשבים והרחיב לי אופקים. אם כי, מנגד, הפרויקט לדעתי היה מאוד ארוך וגם קשה. היו חלקים ואלגוריתמים שלקחו לי שבועות, דבר שכמעט שבר אותי כמה פעמים במהלך הפרויקט. בכל זאת, המשכתי. כנראה שכן היה בי הרצון לעשות את הפרויקט, לנצח אתגר הנראה בעיניי בלתי אפשרי, וכמובן לקבל עוד 5 יח"ל. השקפתי לגבי העולם הטכנולוגי שונתה לחלוטין, התחלתי להבין לעומק איך דברים עובדים ומה ההיגיון מאחורי מוצרים טכנולוגיים רבים בשוק. הרחבת הידע לאט לאט הביאה לי ביטחון בעולם הזה, דבר שגרם לעלייה פרבולית בקצב ההתקדמות והידע שלי. כך אחרי מספר חודשים אני כאן כותב את הסיכום הזה, שהפרויקט שלי ברובו מוכן. אני לוקח איתי להמשך הדרך את כל הידע שצברתי שיעזור לי גם אחרי בית ספר, אם זה בקבלת תפקיד טכנולוגי טוב בצבא שקיבלתי הוא אם זה אח"כ באזרחות. אני מודה, היו לי קשיים שלא חשבתי שאתגבר עליהם וכמה לילות חסרי מנוחה בגלל החשיבה הלא מפסקת על עקיפת הבעיות. אך עדיין כל שבוע ישבתי וניסיתי לעקוף, לפתור איכשהו ואפילו לצמצם בעיות כלשהם. דוגמא לקושי שהיה לי הוא בהקמת השרת הראשוני, בשנה הקודמת לתחילת הפרויקט (כיתה י') לא צברתי ידע מספק לתחילת הפרויקט. לא הצלחתי ולא הבנתי לקוח-שרת, כתיבת הקוד שלי הייתה לוקה בחסר וזה דבר שמאוד הקשה עליי בכתיבה של הבסיס הראשוני של הפרויקט שלי. כחודש ישבתי רק באימון על כתיבת קוד של שרת-לקוח. ואז שדרגתי למודל של שרת-לקוחות וכן הלאה. ללא ספק הייתה לי עלייה גדולה מתחתית מסויימת להצלחה יפה מאוד לטעמי. היו קשיים מייאשים, החלטתי לוותר על רוב הלמדת מכונה בפרויקט שלי לאחר שבועיים של ניסיונות, וזה הציף אותי בשאלות כמו אם אתה לא מצליח לבצע את המרכז בפרויקט שלך איך תעשה את שאר הפרויקט או האם הפרויקט שלי יהיה לא טוב. אבל המשכתי בשלי, וטוב שכך. אני מסתכל אחורנית, ואני מבין שאני יכול לעשות וללמוד המון דברים, מסובכים קשים ומייאשים ככל שיהיו, אם רק יש לי את הרצון. היכולת בי קיימת. זה טוב לאתגר את עצמך כדי להגיע למסקנה כזו, מסקנה שנותנת לך ביטחון להמשיך. אם כי יכולתי לבצע דברים בצורה הרבה יותר נכונה, ושגיתי. עשיתי טעות ש"רצתי" ישר לכתיבת הפרויקט עם שיטה מסויימת במקום לשבת ולקרוא יום יומיים על האופציות האחרות. זה בא לידי ביטוי באופן הכי מובהק בבחירת יצירת מסך GUI עם tkinter. זה היה שיטה נוראית, מסורבלת וקשה שהעסיקה אותי חודשיים בבניית ה-GUI. היו שיטות פי כמה וכמה יותר קלות, טובות ומהירות ובכל זאת מיהרתי. אבל עדיין טעויות כאלה היו בשבילי גם אתגר. כן אם הייתי חושב יותר ופחות כותב קוד הייתי מגיע לתוצאה יותר טובה. אבל עדיין, זה פעם ראשונה שאני בונה אפליקציה משלי וזה לגיטימי שאני אטעה. לסיכום, הפרויקט היה קשה, אך מעצים ואני מרוצה מהתוצאה וברוב הדרך. תלמיד סייבר, כיתה יב', אלמוג גל.